**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**
**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»**

Теплоенергетичний факультет

Кафедра автоматизації проектування енергетичних процесів і систем

«До захисту допущено»
Завідувач кафедри
_____ О.В. Коваль
(підпис) (ініціали, прізвище)
“___”_____2019р.

# Магістерська дисертація

зі спеціальності 121 Інженерія програмного забезпечення
за спеціалізацією Інженерія програмного забезпечення розподілених систем
на тему: Реалізація нейронних мереж в мобільних застосунках

Виконала: студентка 6 курсу, групи ТВ-з81мп
Марич Тетяна Ігорівна _____ _____
(прізвище, ім’я, по батькові) (підпис)

Науковий керівник доцент, к.т.н. Шаповалова С.І. _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Рецензент_____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ - 2019

**Національний технічний університет України**
**«Київський політехнічний інститут ім. Ігоря Сікорського»**

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти другий, магістерський

зі спеціальності - 121 Інженерія програмного забезпечення
за спеціалізацією – Інженерія програмного забезпечення розподілених систем

ЗАТВЕРДЖУЮ
Завідувач кафедри

_____ О.В. Коваль
(підпис)

”___”_____2019р.

# ЗАВДАННЯ
## на магістерську дисертацію студенту

Марич Тетяні Ігорівні
(прізвище, ім’я, по батькові)

1. Тема роботи Реалізація нейронних мереж в мобільних застосунках
керівник роботи Шаповалова Світлана Ігорівна, к.т.н., доцент
(прізвище, ім’я, по батькові науковий ступінь, вчене звання)

затверджена наказом вищого навчального закладу від ”__”___ 201__р. № __
2. Строк подання студентом роботи ____ грудня 2019р.
3. Об’єкт дослідження Програмна реалізація нейронних мереж в мобільному застосунку
4. Предмет дослідження Нейронні мережі бібліотеки TensorFlow Lite
5. Перелік питань, які потрібно розробити провести аналіз технологій для використання нейронних мереж в мобільних застосунках; запропонувати програмне рішення для розв’язання задачі розпізнавання об’єктів на зображенні; підібрати необхідне програмне забезпечення для перекладу тексту; забезпечити збереження даних на пристрої користувача; розробити інтерфейс мобільного застосунку; реалізувати модуль для перетворення інформації у формат навчання.
6. Орієнтовний перелік ілюстративного матеріалу: задачі системи, клієнт-серверна взаємодія з нейронними мережами, клієнтська взаємодія з нейронними мережами, процес інтеграції моделі PyTorch Mobile, архітектура TensorFlow Lite, сімейство MobileNets, Інфраструктура React Native, нативна частина фреймворку, запуск React Native застосунку, архітектура програмного застосунку, REST тип взаємодії з Google Translate API, структура навігації застосунку, структура бібліотеки Redux.

7. Орієнтований перелік публікацій Застосування нейронних мереж в мобільних застосунках (“Сучасні проблеми наукового забезпечення енергетики” XVII міжнародної науково-практичної конференції аспірантів, магістрів, студентів)

7. Консультанти розділів роботи

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | завдання прийняв |
| | | | |

8. Дата видачі завдання ”___” _____ 2019 р.

## КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів виконання магістерської дисертації | Строки виконання етапів магістерської дисертації | Примітка |
|---|---|---|---|
| 1 | Отримання завдання | 28.09.18 р. | |
| 2 | Робота з літературними джерел | 01.10.18 р. – 03.02.19 р. | |
| 3 | Підготовка матеріалів дисертації | 04.02.19 – 31.05.19 р. | |
| 4 | Підготовка доповіді на конференції | 11.03.19 – 29.03.19 р. | |
| 6 | Розробка програмного рішення | 03.06.19 – 25.10.19 р. | |
| 5 | Переддипломна практика | 02.09.19 – 25.10.19 р. | |
| 7 | Захист програмного продукту | 25.10.19 р. | |
| 8 | Розробка стартап-проекту | 11.11.19 – 19.11.19 р. | |
| 9 | Передзахист | 22.11.19 р. | |
| 10 | Оформлення дисертації | 21.11.19- 05.12.19 р. | |
| 11 | Захист | | |

Студент _____ _____Марич Т.І._____
(підпис) (прізвище та ініціали,)

Керівник роботи _____ _____Шаповалова С.І._____
(підпис) (прізвище та ініціали,)

# РЕФЕРАТ

Останніми роками спостерігається інтенсивний розвиток технологій, що стосуються мобільної розробки та роботи з нейронними мережами. Завдяки цьому активно створюються мобільні застосунки, які надають зручний інтерфейс. Нейронні мережі все більше впроваджуються у різні типи систем, вирішуючи задачі класифікації, прогнозування, автоматизації, розпізнавання образів. Сучасні технології дають можливості для роботи з нейронними мережами не тільки на потужних серверах, але і у браузерах та мобільних застосунках.

Метою даної роботи є створення програмного рішення для мобільного застосунку на основі нейронних мереж для вивчення слів іноземними мовами.

Для роботи з нейронними мережами використано фреймворк TensorFlow Lite, який надає програмний інтерфейс для запуску нейронних мереж безпосередньо на пристроях користувачів. Для написання безпосередньо програми обрано фреймворк React Native, завдяки якому можлива одночасна розробка під мобільні платформи Android та iOS.

В ході виконання розглянуто концепцію роботи TensorFlow Lite, підтримувані моделі навчання, інфраструктуру React Native.

Мобільний застосунок забезпечує виконання розпізнавання об’єкту на зображенні, переклад інформації на вибрану іноземну мову, збереження інформації на пристрої користувача.

Для користувача на вибір надається два способи для завантаження зображень: з камери та галереї фотографій.

Наразі застосунок підтримує три мови перекладу: англійську, французьку та іспанську. За необхідності перелік мов може бути розширено. Користувач застосунку має змогу одночасно працювати з декількома мовами, переключаючи їх на екрані налаштувань.

Для вивчення слів створено необхідний функціонал. За бажанням користувач може перейти на екран вивчення, де одне за одним будуть

відображатись зображення, його опис англійською мовою та п’ять варіантів перекладу. Після вибору варіанту відповіді користувач переходить до наступного слова. Кількість вивчень слова та правильна кількість відповідей зберігається для кожного зображення.

Особливістю системи є збереження усієї інформації безпосередньо на пристрої користувача, що забезпечує конфіденційність даних. Таким чином усі зображення, що використовуються в застосунку, залишаються лише в пам’яті пристрою і не зберігаються додатково на зовнішніх серверах.

В результаті роботи створено мобільний застосунок для визначення слів іноземними мовами за фотографією об’єкта

Дана робота містить 95 сторінок, 37 рисунків, 22 таблиці та 33 посилання.

Ключові слова: нейронна мережа, TensorFlow Lite, мобільна розробка, React Native.

# ABSTRACT

In last years there is increasingly intensive technology development related to mobile development and neural networks. The number of mobile applications which provide a user-friendly interface increases every day. Neural networks has being implemented in different systems for solving classification, forecasting, automation, pattern recognition problems. Modern technologies make it possible to work with neural networks not only on powerful servers, but also in browsers and mobile applications.

The purpose of this work is to create a software solution for mobile application based on neural networks for learning words in foreign languages.

Framework TensorFlow Lite, which provides a program interface to run neural networks models directly on users' devices was used to work with neural networks. Framework React Native was chosen to write the mobile application itself, allowing simultaneous development for Android and iOS mobile platforms.

During the work was done review of  concept of TensorFlow Lite, supported models of training and React Native framework.

Mobile application performs object recognition of image, translates information into the selected foreign language, saves information on the user's device.

There are two ways for the user to upload images: from the camera and the photo gallery.

Currently, application supports three translation languages: English, French, and Spanish. If necessary, the list of languages can be expanded. User can work with multiple languages at the same time by switching them on the settings screen.

There was the necessary functionality created learn words. If desired, user can open study screen and see one by one the image, it description in English and five translation options to be displayed. After choosing answer, the user proceeds to the next word. The number of attempts and number of correct answers are stored for each image.

One of the main features of application is the storage of all information directly on the user's device to provide data privacy. In this way, all images used in the application are stored only in the device memory, not on external servers.

Mobile application for defining words in foreign languages by a photo or image is the result of work.

This work contains 95 pages, 37 figures, 22 tables and 33 references.

Keywords: neural network, TensorFlow Lite, mobile development, React Native.

# ЗМІСТ

# ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

1. НМ (з англ. neural networks) — нейронні мережі
2. API (з англ. application program interface) — прикладний програмний інтерфейс
3. REST (representational state transfer) — архітектурний стиль передачі даних у розподіленій системі
4. SLAM (Simultaneous Localization And Mapping) — метод одночасної локалізації та побудови карти

# ВСТУП

За останні декілька років нейронні мережі здобули велику популярність. Це пов’язано з широким спектром їх використання: розпізнавання і обробка об’єктів зображень, розпізнавання та відтворення мови, пошук найкращих пропозицій для клієнтів в веб-застосунках. Широке поширення нейронні мережі отримали і в розробці мобільних додатків. Існує великий спектр їх використання, наприклад, для розпізнавання людей і створення унікальних зображень, в клавіатурі IOS систем, яка видає припущення щодо слова, що набирається. Також деякі пристрої мають систему розпізнавання схожих людей на фотографіях і сортування таких фотографій в окремі папки. Звісно, перелік використання значно ширший. Тому створення архітектурного рішення розв`язання таких задач є актуальним і має практичне значення.

Варто враховувати, що всі ці програми нічому не вчаться безпосередньо у користувача. Це пов’язано з низьким рівнем потужності сучасних мобільних телефонів. Тому найчастіше весь процес навчання відбувається на віддаленому сервері з великою кількістю GPU. Отже, існує два способи використання нейронних мереж в мобільних застосунках: шляхом відправлення запитів на сервер або ж використанням моделі безпосередньо на пристрої. Перший спосіб характеризується постійним навчанням моделі за рахунок запитів користувачів. Наприклад, на стороні клієнта задане вхідне зображення. Це зображення відправляється на сервер, де відбувається аналіз за допомогою моделі та відправлення результату клієнту. Перевагами другого способу є конфіденційність даних, незалежність роботи від наявності інтернету, а в майбутньому - і автоматичне оновлення системи за рахунок самонавчання. Популярним рішенням є використання TensorFlow Lite.

TensorFlow Lite — легковісна бібліотека, орієнтована на розробників мобільних та вбудованих пристроїв для розробки додатків під Android, iOS, Raspberry PI та інших. Найважчою частиною використання бібліотеки є

підготовка готової моделі, а саме конвертація її в модель .tflite. Перевагою даної бібліотеки є набір готових до використання моделей для різних типів задач [1]:

– MobileNet для класифікації зображень, а саме - ідентифікації сотень типів об’єктів;

– PoseNet для розпізнавання пози людини на зображенні або відео;

– Coco-ssd для розпізнавання кількох об'єктів на зображенні з обмежувальними рамками (80 різних класів об'єктів);

– Smart reply для створення пропозицій відповідей на основі повідомлень у чаті;

– DeepLab для сегментації семантичного зображення.

В даній роботі використано бібліотеку TensorFlow Lite для реалізації нейронних мереж на мобільному пристрої.

# 1 ПРОГРАМНА РЕАЛІЗАЦІЯ НЕЙРОННИХ МЕРЕЖ В МОБІЛЬНИХ ЗАСТОСУНКАХ

Розвиток нейронних мереж набуває все більших обертів. Зростає кількість програмних рішень, що надають інструментарій, необхідний для інтеграції НМ у різні типи систем.

В даному розділі описано постановку задачі, розглянуто методи інтеграції нейронних мереж, найвідоміші технології для реалізації нейронних мереж в мобільних застосунках.

Також наведено опис обраної технології для впровадження НМ та приклади програмних рішень, що використовують моделі НМ.

## 1.1 Побудова мобільного застосунку на основі нейронних мереж

Метою даної роботи є створення програмного рішення для мобільного застосунку на основі нейронних мереж для вивчення слів іноземними мовами.

Мета прикладної програмної системи — визначення слів іноземними мовами за фотографією об’єкта.

Задача створення програмного рішення складається з ряду підзадач таких як:

- розпізнавання зображення;
- переклад контенту на різні мови;
- збереження інформації;
- перетворення інформації в формат навчання;
- створення зручного інтерфейсу (для обрання мови, вивчення слів).

Застосунок повинен реалізувати використання нейронних мереж безпосередньо на мобільному застосунку, з можливістю вибору моделі для

застосування та розпізнавання об’єкту на зображенні. Для забезпечення кращих результатів та їх порівняння можливе використання декількох моделей одночасно.

Основним джерелом інформації є зображення, які створені та завантажені користувачем. В застосунку має бути реалізована можливість вибору джерела зображення, а саме: з камери або галереї зображень.

Програмне рішення повинне забезпечувати переклад контенту на декілька іноземних мов з можливістю зміни обраної мови для навчання та збереженням даних відповідно до неї.

Збереження усієї інформації повинне відбуватись тільки на пристрої користувача, тобто в файловій системі та кеші.

Для вивчення слів застосунок має надавати відповідний функціонал з можливістю перегляду статистики щодо вивчених слів та тих, що були використані для навчання.

Необхідний для роботи користувача функціонал наведено на Рис. 1.1 .

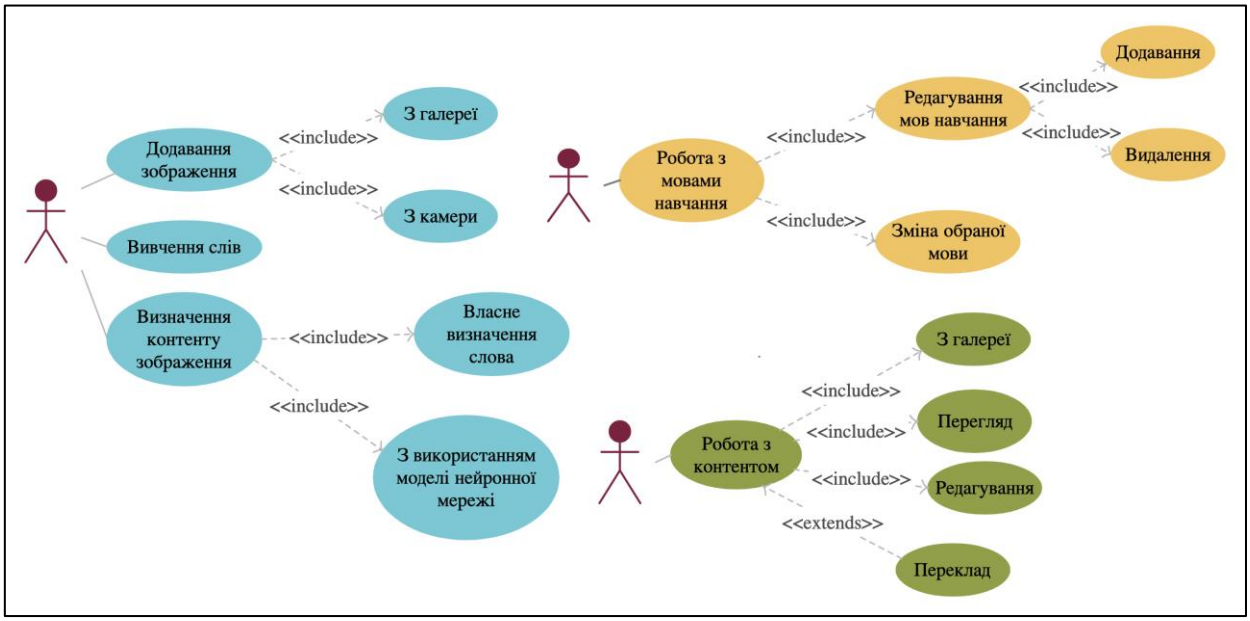


Рис. 1.1 — Задачі системи

Застосунок повинен забезпечити реалізацію зручного, зрозумілого інтерфейсу як для основного функціоналу, так і для налаштувань.

Програмний продукт повинен бути реалізованих на основі даних технологій:

– модульне програмування;

– принципи REST;

– бібліотека для розробки та навчання моделей машинного навчання TensorFlow;

– фреймворк для мобільних додатків React Native;

– Google Translate API для перекладу контенту.

Контент додатку групується за основною спільною характеристикою, а саме кінцевою мовою перекладу.

## 1.2 Методи інтеграції нейронних мереж

Нейронна мережа представляє собою обчислювальну систему, для навчання якої потрібні великі обсяги інформації. Більше того, для отримання кращих результатів для згорткових нейронних мереж необхідна також велика кількість шарів, і, відповідно велика кількість обчислювальних ресурсів [2]. Таким чином виникає проблема забезпечення системи необхідними даними, а також ресурсами для їх зберігання та обробки. Існує два способи роботи з нейронними мережами в мобільних застосунках: клієнт-серверний та клієнтський.

### 1.2.1. Клієнт-серверна взаємодія з НМ

Оскільки для навчання нейронних мереж необхідна велика кількість обчислювальних потужностей та ресурсів, найзручнішим способом використання НМ є її розміщення на сервері. Тому нейронні мережі розміщуються та навчаються розробниками на потужних серверах [3], після чого вже навчена мережа використовується для цілей системи.

Таким чином взаємодія з НМ відбувається шляхом передачі даних клієнтом на сервер, де вони обробляються. Далі сервер повертає результати обробки клієнтові (Рис. 1.2).

Даний спосіб має ряд переваг та недоліків.

Перевагами є:

– наявність необхідних ресурсів;

– навчання нейронної мережі;

– адаптація моделі НМ під запити користувача;

– підстановка оновленої моделі замість існуючої без додаткових конфігурацій на стороні клієнта.

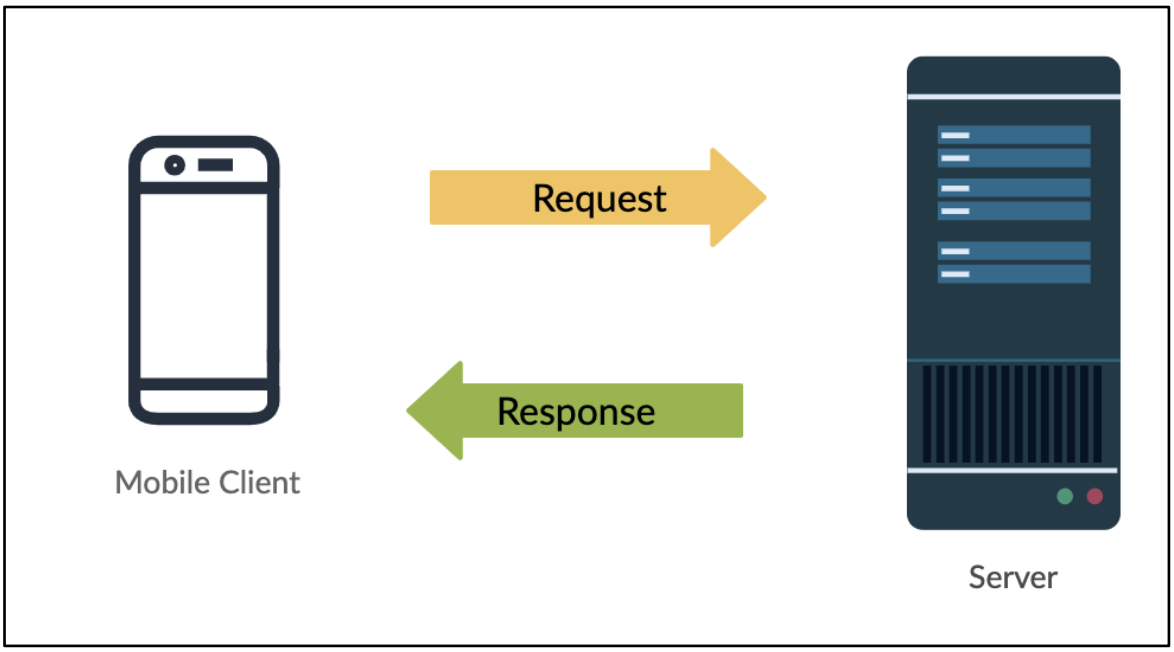


Рис. 1.2 — Клієнт-серверна взаємодія з НМ

Адаптація моделі НМ під запити користувача означає, що відправлені з клієнта дані можуть використовуватись не тільки для їх обробки, але і для тренування моделі НМ [4]. Таким чином з кожним запитом клієнт отримуватиме все точніші результати.

Проте в даного способу існують і недоліки:

– пряма залежність від доступу до інтернету;

– проблеми з сервером відображаються на всіх користувачах;

– проблеми з захистом даних користувачів.

Оскільки взаємодія між клієнтом та сервером відбувається шляхом відправки запитів та отримання відповідей, існує пряма залежність від доступу до інтернету та його швидкості.

Будь-яка помилка на сервері загрожує стабільній роботі клієнтів.

Використання даних для навчання НМ означає необхідність у їх безпечній доставці з клієнта, збереженню на сервері, а також захисту сервера від атак.

Найчастіше розробки надають перевагу даному способу, оскільки не кожен клієнт має необхідну кількість ресурсів або ж здатен забезпечити виконання усіх поставлених до системи вимог [5].

### 1.2.2. Клієнтська взаємодія з НМ

Даний спосіб є відносно новим, оскільки необхідні для даної задачі технології знаходяться у процесі активного розвитку. Клієнтська взаємодія означає використання нейронних мереж безпосередньо на клієнті (Рис. 1.3). Незважаючи на обмеженість ресурсів, необхідність в такому способі виникає все частіше.

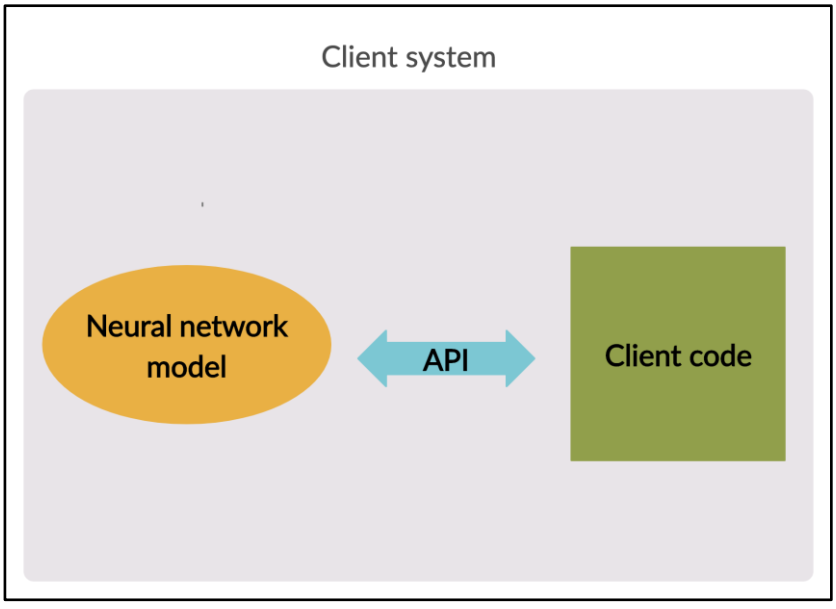


Рис. 1.3 — Клієнтська взаємодія з НМ

Цьому передує ряд особливостей такого типу взаємодії:

– конфіденційність, вся інформація зберігається на пристрої;

– відсутність необхідності доступу до інтернету.

Конфіденційність даних користувача є важливим аспектом розробки будь-якої системи, враховуючи кількість кібератак та кіберзлочинів. Такі атаки загрожують не лише цілісній роботі системи, а й безпеці персональних даних. Тому системи, що працюють з даними користувача безпосередньо на пристрої є безпечнішими.

Оскільки вся необхідна інформація зберігається на девайсі користувача, то відпадає необхідність у стабільному доступі до інтернету.

Проте такий спосіб також має ряд недоліків, серед яких:

– неможливість тренування моделі НМ на клієнті;

– адаптація НМ під дані користувача;

– додаткові конфігурації для роботи з НМ.

Сучасні технології дають можливість використовувати уже навчені моделі на клієнтах. Проте навчання безпосередньо на клієнті — досить складна задача, оскільки потребує виконання великої кількості операцій, а відповідно і ресурсів. Більше того, широкий вибір пристроїв з різними технічними характеристиками ускладнюють цей процес. Незважаючи на це, використання моделей НМ на клієнті є одним з кроків до міграції нейронних мереж з серверів на пристрої. Оскільки відсутня можливість навчання моделей НМ на пристрої за рахунок даних користувача, то відсутня і можливість адаптації НМ під нього.

Робота з НМ на пристрої може бути реалізована шляхом її вбудовування у розроблюваний застосунок. Але така система немає можливості до розширення моделі. Якщо ж необхідно мати можливість оновлювати модель НМ, то це можна зробити шляхом скачування моделі на пристрій і її завантаження в застосунок. Проте такий спосіб також має ряд обмежень, так як потребує додаткової конфігурації при завантаженні оновленої моделі.

На даному етапі розвитку НМ уже з’являються технології, які надають функціонал для навчання моделей НМ безпосередньо на пристроях, проте вони є ще досить обмеженими та сирими. Наприклад, існує платформа Swift for

TensorFlow, яка надає можливість навчання мереж для iOS систем. Проте поки дана платформа надається лише в бета-версії.

## 1.3 Технології для використання нейронних мереж на пристрої

Існує багато інструментів для реалізації нейронної мережі на мобільних пристроях. Серед них Apple BNNS, PyTorch Mobile, TensorFlow і MXNet. Найчастішими ж у використанні є TensorFlow і MXNet.

### 1.3.1. MXNet

Це бібліотека з відкритим кодом, що призначена для навчання та розгортання глибоких нейронних мереж. Вона призначена для роботи в динамічній хмарній інфраструктурі, використовуючи розподілений параметризований сервер і здатна до майже лінійного масштабування за використанням декількох графічних або центральних процесорів.

Дана бібліотека підтримує імперативну та декларативну парадигми програмування, а також розгортання навченої моделі для малоресурсних пристроїв, таких як мобільні пристрої та пристрої інтернету речей.

Бібліотека надає легке і лаконічне API для машинного навчання [6]. Дана система має ряд переваг, а саме:

– простота використання з інтерфейсом Gluon. Бібліотека Gluon MXNet надає високорівневий інтерфейс, який дозволяє легко створювати прототипи, навчати і розгортати моделі глибокого навчання без шкоди для швидкості навчання. Gluon пропонує високорівневі абстракції для визначених шарів, функції втрат і оптимізатори;

– покращена продуктивність. Робочі навантаження можуть бути розподілені між кількома графічними процесорами практично з лінійною масштабністю, що дозволяє системі впоратися за менший час [7];

– для IOT і периферійних пристроїв. Створює спрощені представлення моделей, які можуть працювати на малопотужних периферійних пристроях;

– гнучкість. Підтримує широкий спектр мов програмування, включаючи C++, JavaScript, Python, R, Matlab, Julia, Scala, Clojure і Perl.

Також MXNet підтримується великими постачальниками хмарних послуг, включаючи Amazon Web Services та Microsoft Azure.

### 1.3.2. PyTorch Mobile

PyTorch — це фреймворк для навчання нейронних мереж, створений компанією Facebook. Стабільно набирає популярності в спільноті машинного та глибокого навчання.

З недавнього часу включає експериментальну збірку PyTorch Mobile, новий робочий процес розробки від Python до мобільної розробки. Підтримує повний робочий цикл від Python до деплойменту на iOS і Android системи [8] (Рис. 1. 4).

Дане рішення має ряд переваг, таких як:

– Кросс-платформеність. Підтримка різних мобільних платформ не є тривіальним завданням, оскільки існують значні відмінності в апаратному та програмному забезпеченні iOS та Android систем;

– Квантування для оптимізація моделі. Одним з основних обмежень при розгортанні НМ на мобільних пристроях є розмір моделі. Великі моделі швидко збільшують розмір додатку. Квантування — один із основних методів зменшення розмірів моделі, не впливаючи на продуктивність. Початкова версія PyTorch Mobile підтримує такі методи квантування як квантування після навчання, динамічне квантування та навчання з урахуванням квантування;

– Час виконання: PyTorch Mobile дозволяє розробникам безпосередньо перетворювати модель PyTorch у формат, готовий для мобільних пристроїв, не потребуючи додаткових інструментів. Все відбувається в рамках бібліотеки.

Проте PyTorch Mobile зараз знаходиться в експериментальній стадії, а тому має декілька суттєвих обмежень:

– PyTorch C++ для iOS. На даний момент PyTorch Mobile працює завдяки бібліотеці C++ PyTorch, що створює проблеми для iOS розробників. Swift не може безпосередньо спілкуватися з бібліотекою C++, тому розробникам потрібно використовувати клас Objective-C як міст між C++ та Swift кодом, або ж створювати обгортку C для бібліотеки C++ [9];

– підтримка центрального процесору. Наступні версії матимуть підтримку мобільних графічних процесорів, проте експериментальна збірка обмежена CPU, що обмежує швидкість виводу на пристрої;

– Обмеження API.

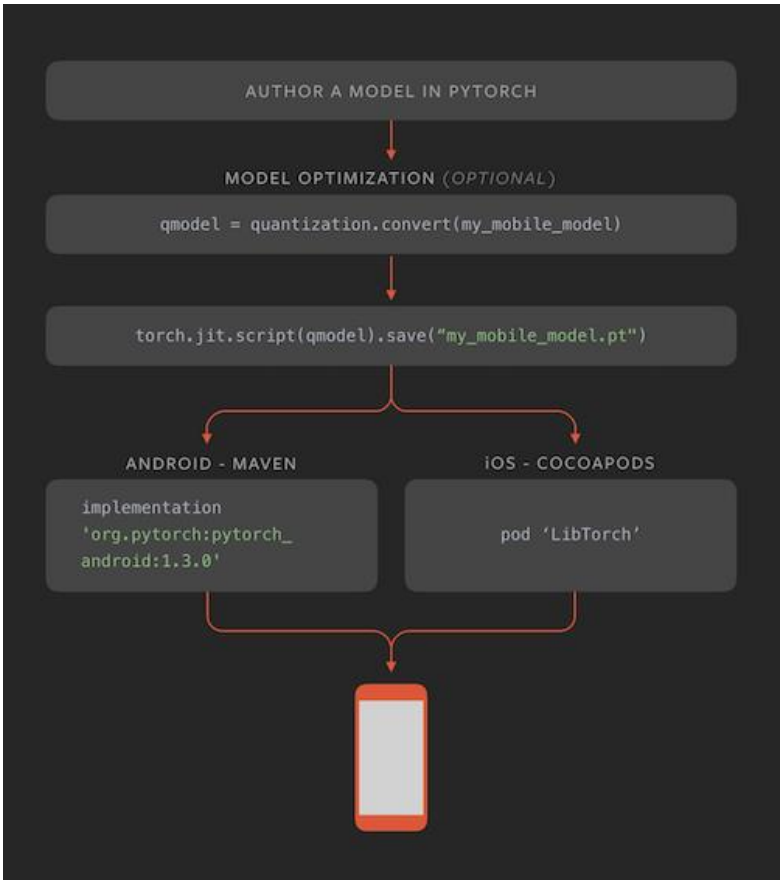


Рис. 1.4 – Процес інтеграції моделі PyTorch Mobile

Дана бібліотека наразі дає досить повільні результати виконання в основному через підтримку лише центрального процесору.

### 1.3.3. TensorFlow

Це відкрита програмна бібліотека для машинного навчання, розроблена компанією Google для вирішення завдань побудови і тренування нейронної мережі з метою автоматичного знаходження та класифікації образів, досягаючи якості людського сприйняття [10]. Особливостями бібліотеки є:

– була створена для обчислення на мобільних пристроях, а також вбудованих системах;

– краще за все спеціалізується на смартфонах;

– спеціально розроблений формат файлу protobuf, що є аналогом JSON в масштабах Google;

– містить техніки для оптимізації та зменшення розмірів моделей.

В результаті аналізу існуючих бібліотек та фреймворків для використання в даній роботі було обрано TensorFlow.

## 1.4 Концепція TensorFlow Lite

TensorFlow Lite ─ це набір інструментів, які допомагають розробникам запускати моделі TensorFlow на мобільних, вбудованих та IoT-пристроях [11]. Завдяки їм на пристроях використовуються малі за розміром моделі, що швидко працюють.

В даний час підтримується на Android та iOS за допомогою C++ API , а також має Java обгортку для Android розробників. Крім того, на Android-пристроях, що підтримують TensorFlow Lite, інтерпретатор також може використовувати Neural Networks API для прискорення апаратного забезпечення, інакше за замовчуванням використовується центральний процесор.

TensorFlow Lite представляє собою середовище виконання, на якому можна запускати вже існуючі моделі, а також набір інструментів, який необхідний для підготовки власних моделей до використання на мобільних та вбудованих пристроях.

На даний момент в TensorFlow Lite немає можливостей для навчання моделей. Проте можливо тренувати моделі на більш потужних машинах, а потім конвертувати їх у формат .tflite, що застосовується в мобільних пристроях [12].

TensorFlow Lite розроблений так, щоб полегшити машинне навчання на пристроях. Більше того, використання нейронних мереж на пристроях за допомогою TensorFlow надає ряд переваг, таких як:

– Латентність: відсутність необхідності відправляти дані на сервер;

– Приватність: дані зберігаються лише на пристрої;

– Підключення: не потрібне підключення до інтернету;

Споживання електроенергії: мережеві з’єднання потребують багато електроенергії.

Бінарний файл важить менше, ніж 300KB .

### 1.4.1. Архітектура TensorFlow Lite

TensorFlow Lite складається з двох основних компонент: інтерпретатора та конвертера. Загальну архітектуру зображено на Рис. 1.5.

Нижче розглянуто кожен компонент детальніше.

TensorFlow Model представляє собою навчену модель TensorFlow, збережену на пристрої.

Дана модель перетворюється за допомогою TensorFlow Lite Converter в формат .tflite, а також виконує оптимізацію задля покращення швидкості взаємодії та розміру файлу [13]. Конвертер TensorFlow Lite (The TensorFlow Lite Converter TOCO) отримує підготовлену модель TensorFlow на вхід і на виході віддає оптимізований для максимальної швидкості і мінімального розміру файл TFLite (.tflite) на основі FlatBuffer, що містить зменшене, бінарне представлення вихідної моделі [14]. Даний файл запускається інтерпретатором в мобільній операційній системі.

Для запуску моделі використовується:

– Java API: C++ API оболонка для ОС Android;

– C++ API: завантажує файл моделі TensorFlow Lite і викликає інтерпретатор [7]. Доступна як на iOS, так і на Android.

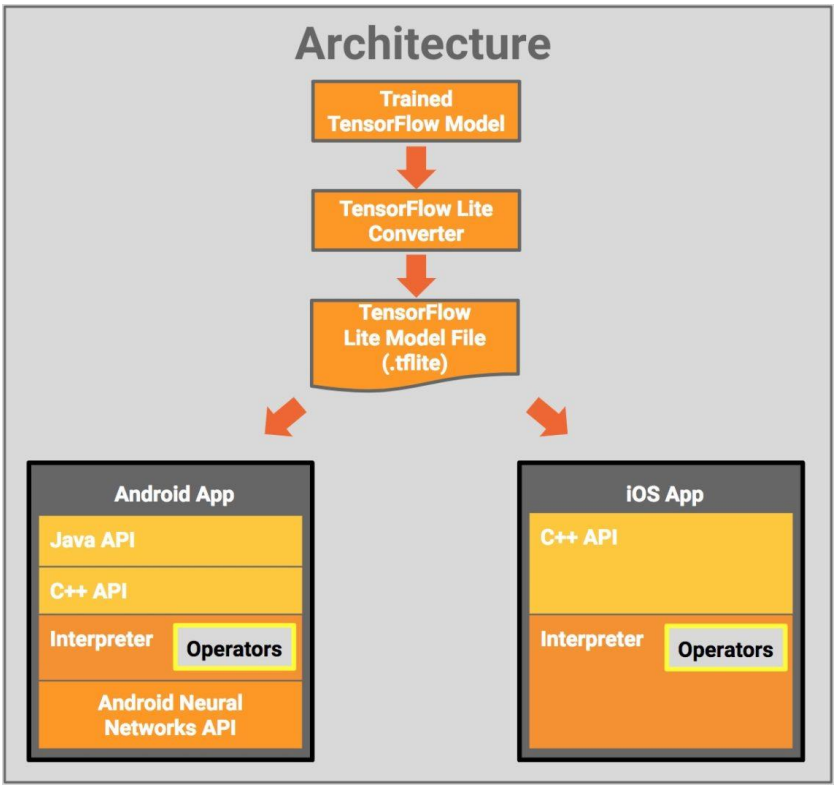


Рис. 1.5 – Архітектура TensorFlow Lite

Інтерпретатор TensorFlow Lite використовує скорочений набір операторів TensorFlow. Обмежуючи оператори, бібліотеки та інструменти за замовчуванням, необхідні для запуску моделей Lite, ядро Інтерпретатора було зменшене до 10000кб без додатків або ~ 300кб з усіма підтримуваними ядрами [15].

### 1.4.2. FlatBuffers

FlatBuffers – це кросплатформна бібліотека для серіалізації мова програмування C ++, C #, C, Go, Java, JavaScript, Lobster, Lua, TypeScript, PHP, Python та Rust. Спочатку він був створений в Google для розробки ігор та інших важливих для продуктивності програм.

FlatBuffers відіграють важливу роль у ефективній серіалізації даних моделі та забезпеченні швидкого доступу до цих даних, зберігаючи невеликий бінарний розмір. Це особливо корисно для файлів моделей, в яких багато числових даних

про ваги, які в силу свого розміру можуть створювати велику затримку в операціях зчитування.

FlatBuffers має ряд переваг, серед яких:

– Доступ до серіалізованих даних без розпакування. Що відрізняє FlatBuffers, це те, що він представляє ієрархічні дані в плоскому двійковому буфері таким чином, щоб до них можна було отримати доступ безпосередньо без розпакування, одночасно підтримуючи еволюцію структури даних (вперед / зворотна сумісність);

– Ефективність і швидкість пам'яті. Єдина пам'ять, необхідна для доступу до даних - це буфер. Для цього потрібно 0 додаткових асигнувань (на C ++). FlatBuffers також дуже підходить для використання з mmap (або потоковою передачею), вимагаючи, щоб у пам'яті була лише частина буфера. Доступ близький до швидкості доступу до необмеженої структури лише з одним додатковим опосередкуванням (різновидом vtable), щоб забезпечити еволюцію формату та необов'язкові поля. Він спрямований на проекти, де витрачати час та простір (багато розподілу пам’яті), щоб мати доступ до або створювати серіалізовані дані, небажано, наприклад, в іграх чи будь-яких інших програмах, що залежать від продуктивності;

– Гнучкість.Необов’язкові поля означають не лише вперед/зворотню сумісність (все важливіше для довготривалих ігор: не потрібно оновлювати всі дані з кожною новою версією), а також вибір способу створення структур даних та даних, які потрібно записувати;

– Крихітний слід коду. Невелика кількість згенерованого коду та лише один невеликий заголовок як мінімальна залежність, яку легко інтегрувати;

– Помилки. Помилки трапляються під час компіляції частіше, аніж під час ручного написання повторюваного коду. Щоб цього уникнути, є можливість автоматично додавати необхідний код;

– Зручний у використанні.Створений C ++ код дозволяє забезпечити швидкий доступ та коду;

– Кросплатформенний код без залежностей. Код C ++ буде працювати з будь-якими останніми gcc / clang та VS2010. Поставляється з файлами збірки для тестів та прикладами (файли Android .mk та cmake для всіх інших платформ) [16].

### 1.4.3. Апаратне прискорення

Оптимізації TensorFlow Lite стосуються також обладнання. Робота в умовах жорстких обмежень пристроїв означає, що процесори повинні використовуватися за високоефективним стандартом.

В Android NDK міститься Neural Network API (NNAPI), яке забезпечує доступ до апаратно-прискорених операцій виводу на Android пристроях. Інтерфейси NNAPI використовують всі можливі апаратні особливості для операторів моделей. З покращенням обладнання переваги NNAPI стануть більш очевидними.

Крім того, 16 січня 2019 року команда TensorFlow випустила підтримку інтерфейсу для GPU, який дозволить певним моделям та операціям вибірково використовувати графічний процесор.

### 1.4.4. Підтримувані моделі навчання

Серед існуючих моделей є п’ять натренованих класів моделей під конкретні типи задач:

– MobileNet – клас моделей комп'ютерного зору для ідентифікації порядку 1000 об'єктів, спеціально розроблений для ефективного виконання на мобільних і вбудованих пристроях [17];

– PoseNet для розпізнавання пози людини на зображенні або відео;

– Coco-ssd для розпізнавання кількох об'єктів на зображенні з обмежувальними рамками (80 різних класів об'єктів);

– Smart reply для створення пропозицій відповідей на основі повідомлень у чаті;

– DeepLab для сегментації семантичного зображення.

В даній роботі було використано модель MobileNets, оскільки вона повністю задовольняє вимогам, а також має велику кількість тренованих для розпізнавання об’єктів.

MobileNets — це сімейство моделей комп’ютерного зору (Рис. 1.6) для мобільних пристроїв для TensorFlow, розроблених для ефективного досягнення максимальної точності, пам'ятаючи про обмежені ресурси для пристрою або вбудованої програми.

Сімейство MobileNets — це невеликі моделі з низькою затримкою та низькою потужністю, параметризовані для задоволення ресурсних обмежень у різних випадках використання.

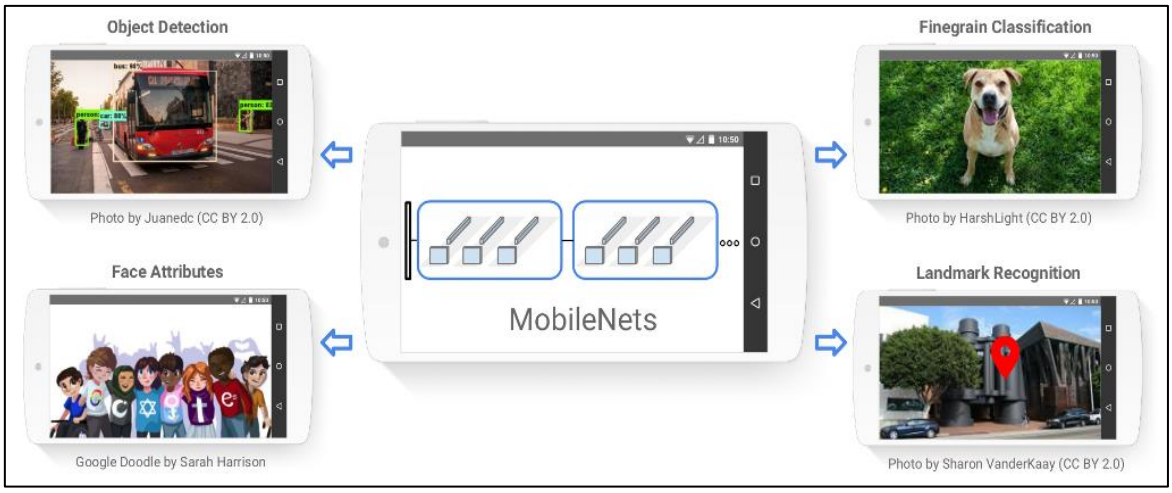


Рис. 1.6 — Сімейство MobileNets

Вони можуть будуватися для класифікації, розпізнавання, вбудовування та сегментації, аналогічно тому, як використовуються інші популярні великомасштабні моделі [12].

### 1.4.5. Впровадження нейронних мереж з TensorFlow Lite

TensorFlow Lite надає всі інструменти, необхідні для перетворення та запуску моделей TensorFlow на мобільних пристроях. Для успішного використання моделі потрібно пройти через декілька етапів її впровадження.

Щоб використовувати модель з TensorFlow Lite, потрібно перетворити повну модель TensorFlow у формат TensorFlow Lite — оскільки не можна створити або навчити модель за допомогою TensorFlow Lite.

Модель можна обрати з уже існуючих в бібліотеці, використати власну або ж потренувати уже існуючу.

Конвертування моделей зменшує розмір файлу і додає оптимізації, що не впливає на точність. Конвертер TensorFlow Lite пропонує варіанти, що дозволяють додатково зменшити розмір файлу та збільшити швидкість виконання, з деякими компромісами. Конвертер представляє собою інструмент, доступний як API Python, який перетворює навчені моделі TensorFlow у формат TensorFlow Lite (Рис. 1.7).

Для запуску моделі використовується Інтерпретатор TensorFlow Lite. Він приймає файл моделі, виконує операції, визначені вхідними даними, та забезпечує доступ до виводу.

```
import tensorflow as tf

converter = tf.lite.TFLiteConverter.from_saved_model(saved_model_dir)
tflite_model = converter.convert()
open("converted_model.tflite", "wb").write(tflite_model)
```

Рис. 1.7 — Конвертація моделі TensorFlow у формат TensorFlow Lite

Інтерпретатор працює на багатьох платформах і надає простий API для запуску моделей TensorFlow Lite з Java, Swift, Objective-C, C ++ та Python.

Також TensorFlow Lite надає інструменти для оптимізації розміру та продуктивності моделей, часто з мінімальним впливом на точність. Оптимізовані моделі можуть вимагати дещо складнішої підготовки, перетворення чи інтеграції.

## 1.5 Приклади програмних систем на основі нейронних мереж

З кожним днем зростає кількість додатків, що використовують НМ в різних сферах. Нижче розглянуто список популярних мобільних застосунків, що використовують НМ для роботи з зображеннями.

### 1.5.1. Prisma

Мобільний додаток, що дозволяє переносити художній стиль на фотографії користувачів, використовуючи нейронні мережі. Перша версія була опублікована в 2016 році та створила справжній ажіотаж. Додаток дає змогу створювати фотокартини на основі 21 витвора мистецтва. Через деякий час після запуску додатку розробники анонсували власний фреймворк для роботи з нейронними мережами безпосередньо на пристроях.

### 1.5.2. FaceApp

Додаток вперше запущений у 2017 році, дозволяє трансформувати зображення, обробляючи його на сервері. Програма на вході отримує портрет людини анфас і надає ряд алгоритмів для старіння, омолодження, зміни статті, кольору волосся, макіяжу людини на фотографії.

### 1.5.3. Dpth

Це додаток, це перетворює 2D зображення у 3D. Початкова ідея полягала у збиранні інформації про дороги для безпілотних машин за допомогою технології SLAM (Simultaneous Localization And Mapping). Проте працюючому алгоритму було досить тяжко зрозуміти відстані до об’єктів, тому розробники вирішили застосувати нейронні мережі.

## Висновки до розділу 1

В даному розділі визначено ряд задач, які необхідно розв’язати для створення програмного рішення:

1. Поставлено задачу визначення слів іноземними мовами за фотографією об’єкта.

2. Визначено складові задачі, реалізація яких потребує готового інструментарію.

3. Описано та визначено спосіб інтеграції нейронних мереж для реалізації поточної прикладної задачі.

4. Досліджено технології реалізації нейронних мереж для мобільних застосунків. Обгрунтовано використання TensorFlow Lite.

5. Розглянуто концепцію TensorFlow Lite.

6. Наведено приклади програмних продуктів для мобільних пристроїв на основі нейронних мереж.

# 2 ПРОГРАМНА РЕАЛІЗАЦІЯ МОБІЛЬНОГО ЗАСТОСУНКУ

В даному розділі реалізацію архітектури програмного рішення, наведено опис застосування обраних технологій.

В першому підрозділі описано інфраструктуру фреймворку React Native, інструменти для роботи з ними та порівняння обраного фреймворку з нативною розробкою.

В другому підрозділі наведено детальний аналіз архітектури програмного рішення, робота з нативними модулями та зовнішнім програмним забезпеченням.

В третьому підрозділі описано реалізацію модуля tflite-react-native, в якому використовуються моделі нейронної мережі на мобільному пристрої для Android та iOS систем за допомогою бібліотеки TensorFlow Lite.

Четвертий підрозділ містить інформацію про обране програмне забезпечення для перекладу слів, а також послідовні кроки його інтеграції в застосунок.

В п’ятому підрозділі наведено опис бібліотек для роботи з кешом та файловою системою мобільного застосунку, а також навігацією.

## 2.1 Фреймворк React native

React Native — це фреймворк, що дозволяє створювати мобільні додатки, використовуючи лише JavaScript. Він використовує той же дизайн, що і React, що дозволяє створювати багатий мобільний інтерфейс користувача з декларативних компонентів.

React Native не використовує WebView — компонент платформи Android, а отже не створює гібридні та веб-додатки. Даний фреймворк дозволяє створити нативну мобільну програму, яка не відрізняється від програми, створеної за допомогою Objective-C або Java [18].

У кожній програмі React Native працюють два важливих потоки.

Один з них – UI Thread, який працює в кожному стандартному нативному додатку. Він керує відображенням елементів користувацького інтерфейсу і обробляє кліки користувача.

Інший — JS Thread, є характерним для React Native. Його завдання - запустити код JavaScript в окремому движку JavaScript [19]. JavaScript працює з бізнес-логікою програми. Він також визначає структуру і функціональні можливості користувацького інтерфейсу. Ці два потоки ніколи не зв'язуються безпосередньо і ніколи не блокують один одного. Між цими двома потоками знаходиться так званий міст, який є ядром React Native.

Міст має такі характеристики:

– Асинхронний. Дозволяє асинхронний зв'язок між потоками. Це гарантує, що вони ніколи не заблокують одне одного;

– Регулювання. Передає повідомлення з одного потоку в інший оптимізованим чином;

– Серіалізація. Обидва потоки ніколи не діляться або працюють з одними і тими ж даними [20]. Замість цього вони обмінюються серіалізовананими повідомленнями.

Також існують ще два потоки, які запускаються не завжди. Native Modules Thread використовується, коли додатку необхідно отримати доступ до мобільної платформи. А також Render Thread, який використовується тільки для деяких версій платформи андроїд та викликає спеціальні команди для відображення інтерфейсу користувача.

Оскільки ігнорувати відмінності між двома мобільними платформами досить важко, іноді необхідно писати специфічний для платформи код. Команда react native пропонує 2 варіанти: модуль Platform та окремі компоненти [21].

В першому випадку потрібно в файлі підключити модуль Platform з пакету react native. Об’єкт даного модуля містить поле “OS”, що повертає назву мобільної платформи, на якій запускається даний проект. Інший варіант застосовується в випадку, коли необхідно застосувати різні компоненти для кожної платформи. Наприклад, компонент Picker на IOS та Android платформах

має зовсім різний вигляд, а отже є платформо залежним компонентом. В такому випадку необхідно розділити код на окремі файли з розширенням android та ios.

React Native автоматично визначає розширення файлу та використовує необхідний компонент в залежності від платформи під час запуску мобільного додатку.

### 2.1.1. Інфраструктура React Native

Фреймворк React Native має досить складну інфраструктуру, що забезпечує виконання JS коду на мобільних пристроях. В більшості ресурсів в основному згадується JS Thread, UI Thread і Bridge, тобто міст, який зв'язує JS і нативний код. Та насправді структура значно складніша [22]. З нею можна ознайомитись на Рис. 2.1.

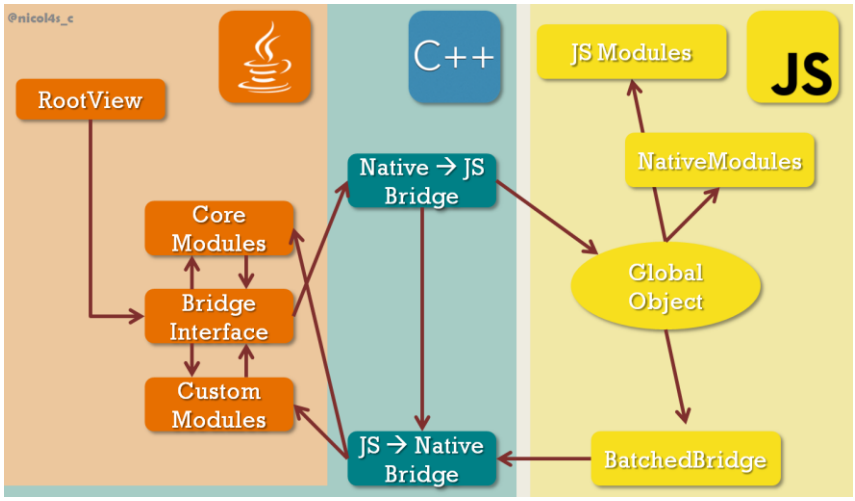


Рис. 2.1 — Інфраструктура React Native

Отже, будь-яка програма має дві складові — це безпосередньо код програми та Main thread, який ще часто називають UI thread. Кодова база містить дві частини — код фреймворку та користувацький код, що містить реалізацію програми. Відповідно кожна з цих частин може бути реалізована як нативною, так і JS мовами, таким чином кодова база складається з чотирьох частин. Під нативною мовою мається на увазі мова програмування, що використовується для написання програм під конкретну платформу.

### 2.1.2. Створення нативних елементів

Кожна компонента, яку фрейморк надає, в кінці кінців буде рендеритись на нативній частині. Цьому відповідає два кроки: створення нативних елементів та та прив'язка їх до компонент JS, а також збереження елементів та їх відображення. UIManagerModule відповідає за перший крок, а RootView — за наступний. RootView представляє собою певний контейнер, що відповідає за збереження всіх нативних елементів, які утворюють дерево — нативну репрезентацію дерева JS компонент (Рис. 2.2).

Як тільки застосунок ініціалізується, відразу створюється пустий контейнер RootView. Далі робота з даним контейнером ведеться через Bridge Interface [23].

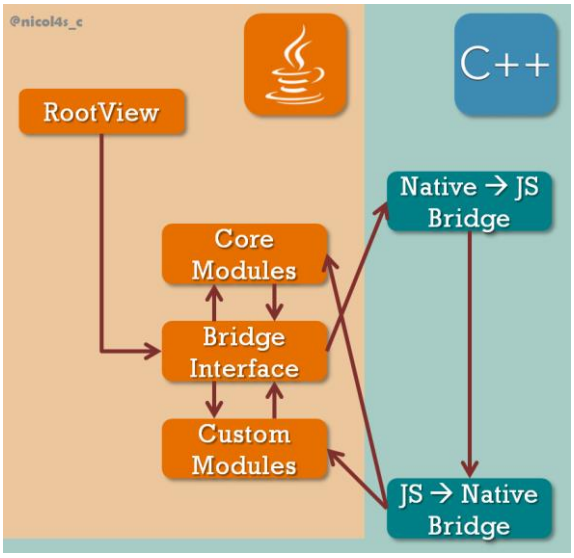


Рис. 2.2 — Нативна частина фреймворку

Bridge Interface постає своєрідним API, що забезпечує сумісну роботу нативного і JS коду. Сам Bridge реалізований мовою програмування C++ і представляє дві складові: своєрідний міст(або зв'язок) від нативного коду до JS і навпаки. Проте сам по собі міст нічого не представляє собою без ендпоінтів (endpoint), до яких можна звертатись. Цими ендпоінтами є нативні модулі (Native Modules), а також Core Modules — єдина частина фреймворку доступна з JS коду. UIManagerModule є складовою Core Modules [24].

При ініціалізації програми для кожного модуля створюється власний екземпляр, посилання на який зберігається на частині мосту, що йде від JS до нативного коду. Завдяки цьому можна працювати з екземпляром з JS коду. А всередині модуля зберігається посилання на Bridge Interface, що дозволяє нативному коду викликати JS код. Після цього буде створено два додаткових потоки: JS Thread і NativeModules Thread (для iOS платформи останній представляє собою декілька потоків).

Вище була розглянута нативна частина фреймворку. Після того, як вся робота на нативній частині виконана, фреймворк починає завантажувати Javascript bundle, що містить код фреймворку та код програми.

### 2.1.3. Віртуальна машина Javascript

Оскільки Javascript є інтерпретованою мовою програмування, весь код написаний на Javascript повинен бути попередньо конвертований в байткод(bytecode) і тільки після цього може виконуватись. За конвертацію та виконання відповідає віртуальна машина Javascript, яку також називають Javascript engine. Таких віртуальних машин існує безліч, включаючи Chrome’s V8, Mozilla’s SpiderMonkey і Safari’s JavaScriptCore. В процесі дебагу React Native використовує V8, а інакше ─ JavaScriptCore [25]. Оскільки JavaScriptCore не підтримується Android платформами за замовчуванням, React Native вбудовує його копію в Android застосунок. Тому Android застосунки важать більше за свої аналоги для iOS.

Перед тим, як запустити JavaScript віртуальну машину, React Native повинен ініціалізувати Context, що представляє собою середовище виконання. Context містить Javascript Global Object, який зберігається на C++ bridge. Завдяки цьому глобальний об’єкт доступний не лише з середовища JavaScript, а й за його межами. Отже, це основний засіб зв'язку між C++ та Javascript, оскільки саме через глобальний об'єкт деякі види нативних функцій стануть доступними для Javascript ─ функції, які, в свою чергу, будуть використовуватися для передачі даних з Javascript у нативну мову.

У глобальному об’єкті зберігається багато речей, але особливо важливими є масив ModuleConfig та функція flushQueue (). Кожен елемент масиву ModuleConfig описує Native Module (будь то Core або Custom), включаючи його ім'я, експортовані константи, методи. Функція flushQueue() відіграє найважливішу роль у забезпеченні зв'язку між Javascript та нативним середовищем для передачі викликів між ними.

Після того, як контекст Javascript був повністю створений і заповнений, він подається в віртуальну машину Javascript, яка починає завантажувати React Native Javascript bundle на JS Thread.

### 2.1.4. Завантаження Javascript bundle

Коли віртуальна машина почне обробляти частину Javascript коду фреймворку, вона створить BatchedBridge (з англ. batch — партія, порція), що представляє собою просту чергу, яка зберігає виклики з Javascript на нативну частину. Виклик — це об'єкт, що містить ідентифікатор нативного модуля, ідентифікатор методу для вказаного нативного модуля, а також аргументи, з якими повинен викликатися нативний метод. Періодично (кожні 5 мілісекунд за замовчуванням) BatchedBridge буде викликатись на global.flushQueue (), передаючи його вміст - масив викликів — з Javascript на іншу сторону C++ bridge[26]. Кожна порція масивів викликів індексується, щоб забезпечити одночасне виконання всіх змін інтерфейсу (це необхідно, оскільки весь процес є асинхронним). Javascript-НК(нативний код) міст, пройдеться по кожному виклику у порції і направить його у відповідний нативний модуль, використовуючи вказаний ідентифікатор модуля.

Наступним кроком є створення об’єкту NativeModules — це об’єкт, який потрібно імпортувати з react native кожного разу, коли необхідно викликати нативний модуль. Об'єкт NativeModules буде заповнений за допомогою згаданого раніше масиву ModuleConfig. Приблизно еквівалентно виконанню NativeModules[module_name] = {} для кожного імені модуля, що міститься у ModuleConfig, тоді NativeModules[module_name][method_name] = fillerMethod для

кожного зовнішнього (або екпортного) методу обраного модуля. fillerMethod просто зберігає всі аргументи, які він отримує в BatchedBridge, поряд із методом та ідентифікатором модуля (fillerMethod = function(...args) { BatchedBridge.enqueueNativeCall(moduleID, methodID, args)}), ефективно створюючи виклик з Javascript на нативну частину. Цей виклик виглядає таким чином з JS коду: MyNativeModule.myMethod (args).

Після цього залишається створити основні JS-модулі, серед яких DeviceEventEmitter, що використовується для передачі подій(events) з нативної частини до Javascript. Javascript містить AppRegistry, який зберігає посилання на основні компоненти додатку. Для того, щоб викликати їх з нативної частини, ці модулі реєструються у глобальному об'єкті (Рис. 2.3).

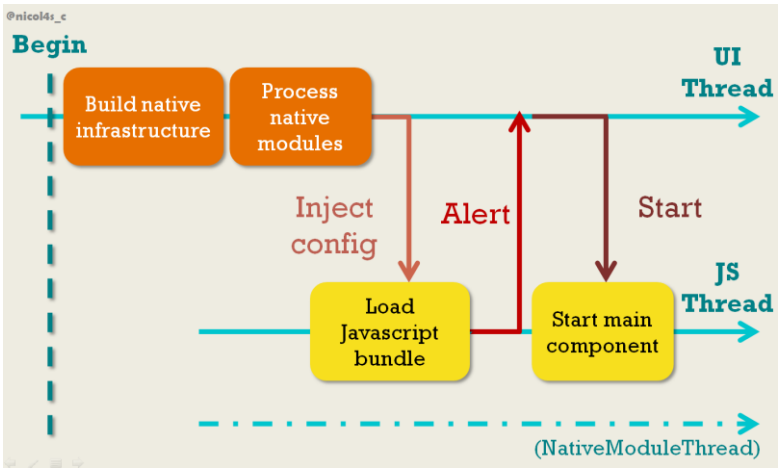


Рис. 2.3 – Запуск React Native застосунку

Завантаження Javascript bundle відбувається на JS Thread, який не залежить від основного потоку (UI Thread). Таким чином, JS Thread повинен попереджати головний потік про виконання свого завдання, а у відповідь головний потік використовує AppRegistry (модуль JS), щоб сказати JS Thread обробити головний користувальницький компонент - зазвичай App.js.

Отже, коротко описавши, фреймворк буде проходиться кожен раз по дереву компонентів Javascript, що міститься в головному компоненті програми, викликаючи UIManagerModule кожного разу, коли трапляється UI компонент [27].

У свою чергу, UIManagerModule (у UI Thread) піклується про створення нативних елементів та збереження їх у RootView.

### 2.1.5. Інструменти React Native

Для спрощення роботи з React Native створено широкий спектр інструментів. Деякі з них є необхідними для комфортної розробки, інші ж варто використовувати лише в конкретних випадках.

React Native має вбудовану можливість підключення до інструментів Chrome Dev Tools, таким чином існує можливість відлагодження проекту, навіть мобільного. Неважливо, запущено додаток в симуляторі чи на фізичному пристрої, в будь-якому випадку існує можливість використовувати Dev Tools в той же спосіб, що і для веб-версій проектів. DevTools допомагають швидко редагувати код та знаходити проблеми, що в кінцевому підсумку допомагає швидше розробляти продукти. Серед основних характеристик, наданих даним інструментом є такі:

- робота з DOM вузлами (редагування атрибутів вузла, його типу);
- перегляд та редагування CSS стилів;
- відлагодження Javascript коду;
- перегляд логів в консолі;
- оптимізація коду;
- робота з мережею (запити та відповіді).

Наступним інструментом є Node.js ─ середовище виконання JavaScript, завдяки якому написано багато інструментів React Native. Крім цього, необхідно використовувати менеджер пакетів Node: npm або yarn, для встановлення деяких інших бібліотек.

Іншим важливим інструментом є Watchman ─ це інструмент з відкритим кодом, створений Facebook. Даний інструмент слідкує за змінами в коді, які він автоматично збирає і після цього автоматично перевантажує симулятор. Для розробника це означає, що не потрібно під час розробки в ручному режимі перезавантажувати проект, що значно прискорює процеси.

Інструмент Xcode ─ основний інструмент для розробників iOS. Зазвичай він використовується для запуску мобільного додатку в iOS Simulator. Проте коли застосунок повинен забезпечити інтеграцію з Facebook, Google та іншими бібліотеками, основна частина конфігурації вимагає налаштування змін в проекті iOS Xcode.

Android Studio ─ це інструмент для нативних розробників Android. Його необхідність зумовлена всіма тими ж причинами, що і Xcode.

Оскільки Xcode і Android Studio є досить важкими програмами, особливо для одночасного виконання, іноді розробники для запуску симуляторів надають перевагу Genymotion для Android симуляторів.Genymotion також дозволяє імітувати багато популярних Android-пристроїв, тому це найпопулярніший вибір серед розробників React Native.

Наступним інструментом є Expo. Це інстумент, що складається з двох частин: XDE та клієнт Expo. XDE ─ це інструмент для створення проектів, перегляду журналів, швидкого попереднього перегляду програм на пристрої, публікації та багатьох інших функцій. Клієнт Expo дозволяє відкривати проекти на телефоні під час роботи над ними, що дозволяє повністю забути про Android Studio або Xcode.

React Native CLI ─ це невелика програма Node, яка пропонує просту команду init, яка використовується для створення нових додатків React Native. Використовуючи React Native CLI, можна створити стандартний додаток React Native з усіма необхідними файлами та залежностями, необхідними для створення додатка для iOS та Android, і можливістю зразу ж відкрити проект за допомогою Xcode або Android Studio [28].

В процесі розробки різних завдань буває необхідним протестувати їх виконання перед релізом версії. Для iOS систем дана вимога задовольняється завдяки Testflight. Даний інструмент дозволяє запрошувати користувачів тестувати програми та збирати цінні відгуки, перш ніж випускати додатки в App Store. Можливо запросити до 10 000 тестувальників, використовуючи лише електронну адресу або поділившись загальнодоступним посиланням.

Якщо в проекті використовується Redux, React Native Debugger ─ це інструмент для відлагодження коду. Це окремий додаток для настільних ПК, який працює на mac OS, Windows та Linux. Він має всі характеристики, наявні у Chrome DevTools і навіть більше. Він об’єднує як DevTools, так і інструменти для розробників Redux в одному додатку, тому немає необхідності працювати з двома окремими програмами для налагодження.У ньому є інтерфейс для DevTools Redux, де можна побачити журнали та усі дії Redux. Це захищає розробника від написання додаткового коду для ручного відлагоджування в DevTools [29].

Наступним інструментом, що значно спрощує та прискорює роботу розробників є fastlane. Даний інструмент надає найпростіший спосіб автоматизувати бета-версію та версії для додатків на iOS та Android. Він справляється з усіма стомлюючими завданнями, такими як створення скріншотів, робота з підписанням коду та релізом програми.

Fastlane містить численні інструменти, серед яких match, gym, pilot, supply.

Наступний інструментом є Metro, що використовується для збірки коду JavaScript. Він отримує вхідний файл, а також різні параметри та повертає один файл JavaScript, який включає весь код та його залежності.

Metro проходить через три етапи в процесі збірки:

– резолюція;
– трансформація;
– серіалізація.
– резолюція

Metro будує графік залежностей усіх модулів, необхідних від точки входу. Щоб знайти усі залежності, Metro використовує розпізнавачі. Насправді цей етап відбувається паралельно зі стадією трансформації.

На етапі трансформації всі модулі проходять через трансформатор, який відповідає за перетворення (транслідування) модуля у формат, зрозумілий цільовій платформі (наприклад, React Native). Трансформація модулів відбувається паралельно залежно від кількості наявних ядер машини.

Після перетворення даних їх необхідно серіалізувати. Серіалізатор поєднує модулі для генерації одного або декількох пакетів. Пакет — це буквально сукупність модулів, об'єднаних в один файл JavaScript.

Отже Metro розбито на декілька модулів, кожен з яких відповідає певному етапу. Дані модулі можна заміняти в залежності від потреб.

### 2.1.6. Порівняння React Native та Real Native платформ

React Native — це фреймворк, створений компанією Facebook для побудови кросплатформенних додатків мовою програмування Javascript.

Real Native платформи реалізують мобільні додатки з використання класичних мов програмування для мобільної розробки, наприклад Java або Swift.

Основною перевагою react native є те, що в результаті розробки створений додаток використовує нативні елементи. Таким чином, правильно спроектований та детально реалізований додаток на react native не буде відрізнятись за виглядом від нативного. Проте зовнішній вигляд додатку є не єдиним критерієм оцінки системи, тому важливо враховувати завантаженість CPU, GPU, а також використання пам`яті.

Завантаженість CPU може варіюватись в залежності від сторінки додатку, проте обидві платформи показали приблизно однаковий відсоток завантаженості. Вимірювання CPU показують, що Swift системи володіють кращими результатами відносно react native. Щодо рівня використання пам`яті, тут однозначним лідером виявився фреймворк react native.

Таким чином, не можна однозначно визначити засіб розробки, що краще справляється з своєю задачею.

Проте у react native є деякі переваги над розробкою на Swift або Java.

По перше, це повторне використання коду. Один і той же написаний код буде працювати як для обох мобільних платформ, так і для веб платформи. Таким чином значно скорочується загальний час розробки додатку. Більше того, така особливість дає можливість створювати певну базу уже реалізацій компонентів, що може використовуватись в різних системам.

Іншою перевагою є набір інструментів з коробки. React native пропонує допоміжні засоби для Android та IOS, додаткові анімації, а також підтримку flexbox для стилізації користувацького інтерфейсу.

З іншого боку, у даної платформи існують і суттєві недоліки. До них можна віднести обмежений api. Фреймворк підтримує велику кількість модулів, написаних на нативній мові, проте часто виникає потреба використання api, що ще нереалізовано. Проте існує можливість самостійного підключення різних модулів.

Ще одним недоліком є відмінності у принципах проектування для IOS та Android. А тому іноді виникає потреба в підтримці розгалуженої системи, що веде себе по-різному в залежності від платформи.

## 2.2 Архітектура програмного застосунку

Відповідно до обраних програмних засобів архітектура програмного застосунку зображена на Рис. 2.4.

Отже, сама система складається з основної частини, написаної мовою програмування Javascript та нативної частини.

В нативній частині коду, а саме в модулі tflite-react-native реалізовано роботу з моделлю нейронної мережі. Даний модуль написано з використанням C++ API фреймворку TensorFlow Lite. Він забезпечує завантаження моделі, передачу обраного зображення у відповідно обрану модель та отримання результатів їх обробки.

Основною кодовою базою проекту є javascript код, що відповідає за реалізацію логіки, інтерфейсу та роботи з різними модулями. Умовно весь код можна поділити на шість частин:

- Сервіси;
- Файли з допоміжними функціями (Helpers);
- Файли для роботи зі збереженням даних (Redux store);
- Навігація;

– Екрани (Screens);

– Компоненти (Components).

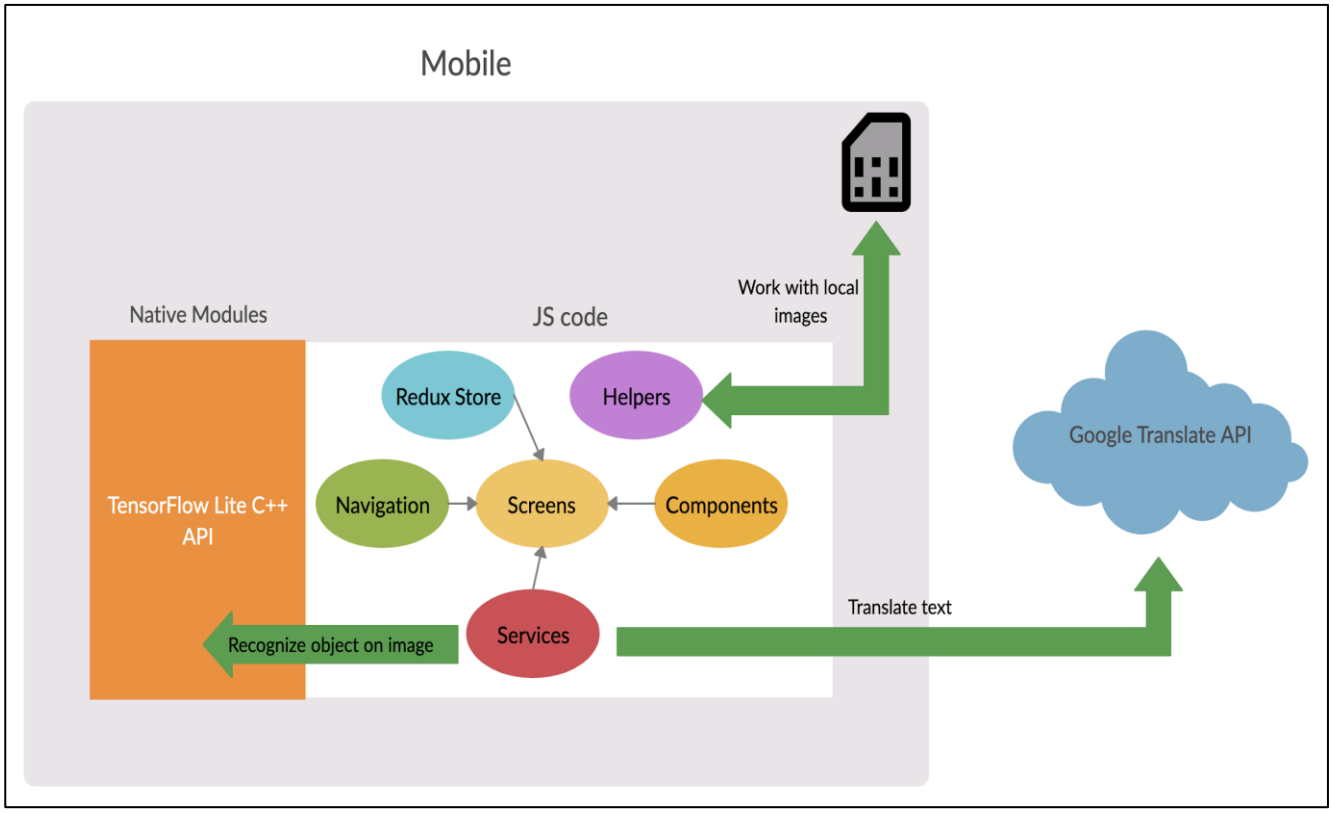


Рис. 2.4 – Архітектура програмного застосунку

Сервіси складаються з двох файлів: googleTranslationApi.js та recognizeService.js.

Перший файл реалізує функцію для перекладу даних (слів) шляхом відправки запиту на Google Cloud Platform та обробкою отриманих результатів.

Другий файл відповідає за виклики з Javascript коду нативних функцій для завантаження моделі НМ та розпізнавання об’єкту на зображенні. Також в даному файлі містяться шляхи збереження застосовуваних моделей та функції для розпізнавання з MobileNet, SSD та YOLO.

У папці Helpers знаходяться два файли: imagePicker.js та storeHelper.js . Пікер містить функцію для відкриття галереї фотографій користувача та функціонал для збереження завантажених зображень у файловій системі користувача. Останнє необхідно для доступу до зображення з застосунку, а також

для випадків, коли користувач видаляє зображення з галереї. StoreHelper.js містить допоміжну функцію для обробки даних в кеші.

Файли папки Redux Store відповідають за більшу частину логіки застосунку, а також збереження усіх необхідних даних у кеші. Тут зберігаються селектори для отримання даних, функції для роботи з сховищем даних застосунку.

У Navigation містяться всі файли, необхідні для реалізації різних типів навігацій. В даному застосунку навігація складається з основної бокової навігації, яка містить три стек навігації. Окремою стек навігацією є ланцюжок екранів при першому завантаженні застосунку.

Папка Screens містить усі екрани, які можуть бути відкриті в застосунку. На кожному екрані зберігається уся логіка, що відноситься до даної частини застосунку, виклики селекторів та зв’язок з сховищем даних застосунку.

В свою чергу кожен екран містить ряд компонентів, що реалізують інтерфейс програми, а також виклики функцій. Всі компоненти зберігаються у папці Components.

## 2.3 Реалізація модуля tflite-react-native для роботи з TensorFlow Lite

Для роботи з моделями було використано модуль tflite-react-native, написаний на C++ .

Першим кроком є завантаження моделі, представленої двома файлами з розширеннями tflite і txt.

Спочатку необхідно завантажити файл tflite, вказавши шлях до нього у FlatBufferModel::BuildFromFile функції. Після успішного завантаження необхідно завантажити txt файл. Далі модель готова до роботи. Всі операції приховані за API модуля, а саме під функцією loadModel.

Наступним етапом є завантаження та обробка зображення. Для цього у функцію runModelOnImage передається ряд параметрів, серед яких шлях до файлу

та додаткові настройки. Всередині дана функція містить виклик функції feedInputTensorImage, що розбиває зображення на тенсори, після чого результат операції використовується інтерпретатором для отримання результатів розпізнавання.

Для успішної роботи модулем tflite-react-native необхідно виконати правильно установку в проект. Будь-який модуль в react native, що містить нативний код, повинен бути прилінкований. Існує два способи: автоматичний та ручний.

Даний модуль потребує саме ручного лінкування, оскільки після автоматичної проект не збирається. Для початку необхідно відкрити файл з розширенням .xcodeproj в ide XCode. Після чого натиснути правою кнопкою мишки на папку Libraries і імпортувати в неї файли з папки node_modules файл TfliteReactNative.xcodeproj у папці ios модуля tflite-react-native. Далі необхідно обрати власний проект, натиснувши на нього та відкрити вкладку Build Phases. Після цього найти в проекті TfliteReactNative в папці Products файл libTfliteReactNative.a та скопіювати його в на обрану вкладку в розділ Link Binary with Libraries. Модуль успішно прилінкований та готовий для використання з Javascript коду.

Для роботи з модулем в Javascript файлах необхідно його спочатку імпортувати (Рис. 2.5).

```
import Tflite from 'tflite-react-native';

let tflite = new Tflite();
```

Рис. 2.5 — Імпорт модуля tflite-react-native

Наступним кроком є завантаження моделі (Рис. 2.6). Для цього у функцію виклику необхідно передати шлях до обраної моделі та її ярлики. Після чого модель успішно завантажена, а тому готова до використання.

```
tflite.loadModel({
  model: 'models/mobilenet_v1_1.0_224.tflite',// required
  labels: 'models/mobilenet_v1_1.0_224.txt',  // required
  numThreads: 1,                              // defaults to 1
},
(err, res) => {
  if(err)
    console.log(err);
  else
    console.log(res);
});
```

Рис. 2.6 — Завантаження моделі

Для обробки зображення викликається функція detectObjectOnImage (Рис. 2.7).

В параметрах функції необхідно вказати шлях до зображення та тип моделі обраної моделі.

```
tflite.detectObjectOnImage({
  path: imagePath,
  model: 'SSDMobileNet',
  imageMean: 127.5,
  imageStd: 127.5,
  threshold: 0.3,       // defaults to 0.1
  numResultsPerClass: 2,// defaults to 5
},
(err, res) => {
  if(err)
    console.log(err);
  else
    console.log(res);
});
```

Рис. 2.7 — Виклик  функції розпізнавання моделі

Кожна функція для роботи з моделлю НМ повертає два параметри: err та res. За успішного виконання параметр err є пустим, але завжди потрібно реалізувати код для обробки помилок.

## 2.4 Робота з Google Translate API

Google Cloud Platform — це набір хмарних служб, що надаються компанією Google.

Однією з таких служб є Google Translate API - інструмент для перекладу тексту на сотні мов. Існує дві версії даної служби: стандартна та розширена.

До стандартної служби входять функції перекладу тексту, визначення мови тексту та отримання переліку підтримуваних мов.

Розширена версія включає створення власного словника, переклад великих обсягів тексту в оффлайн режимі та використання AutoML Translation models для перекладу [30].

Перш, ніж розпочати взаємодію з Google Translate API необхідно створити проект в Google APIs Console вказавши в формі створення назву проекту. Для цього потрібно мати активний Google аккаунт.

Після успішного створення у вкладці Бібліотека необхідно знайти та підключити відповідні API та сервіси (Рис. 2.8). В даному випадку пошук виконується за запитом Google Translate API.

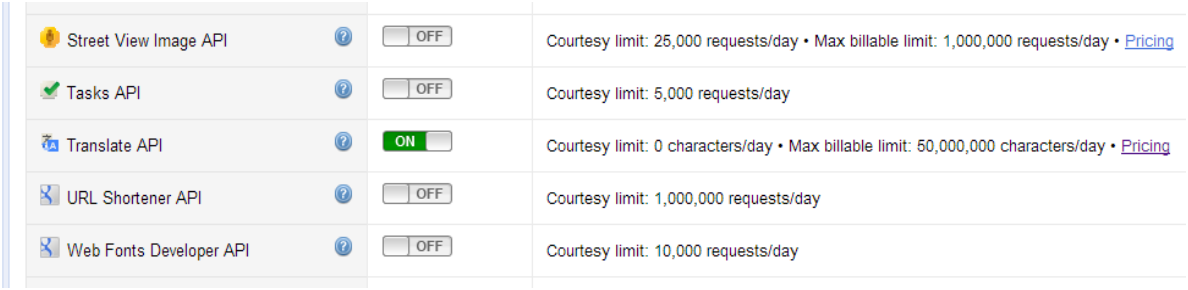


Рис. 2.8 Підключення Google Translate API

Далі у вкладці Облікові дані необхідно створити ключ API, який буде відправлятись одним з параметрів у запиті. Щоб отримати даний ключ, потрібно обрати опцію меню API Access в Google APIs Console.

Існують клієнтські бібліотеки Cloud Translation, доступні для таких мов програмування [16]:

– C#;
– Go;
– Java;
– Node.js;

– PHP;

– Python;

– Ruby.

Оскільки Google офіційно не підтримує жодного модуля для роботи з API через react native, для даної роботи було обрано REST тип взаємодії (Рис. 2.9).

Тобто взаємодія застосунку з API відбувається шляхом відправки запитів в Google Cloud та обробки відповідей. Таким чином за адресою https://translation.googleapis.com/language/translate/v2 відправляються усі дані для перекладу даних, введених юзером або розпізнаних моделлю [32].

Обов'язковим поле є key, куди записується ключ API згенерований в Google Cloud Platform. В тілі запиту необхідно вказати параметри source, target, format і q.

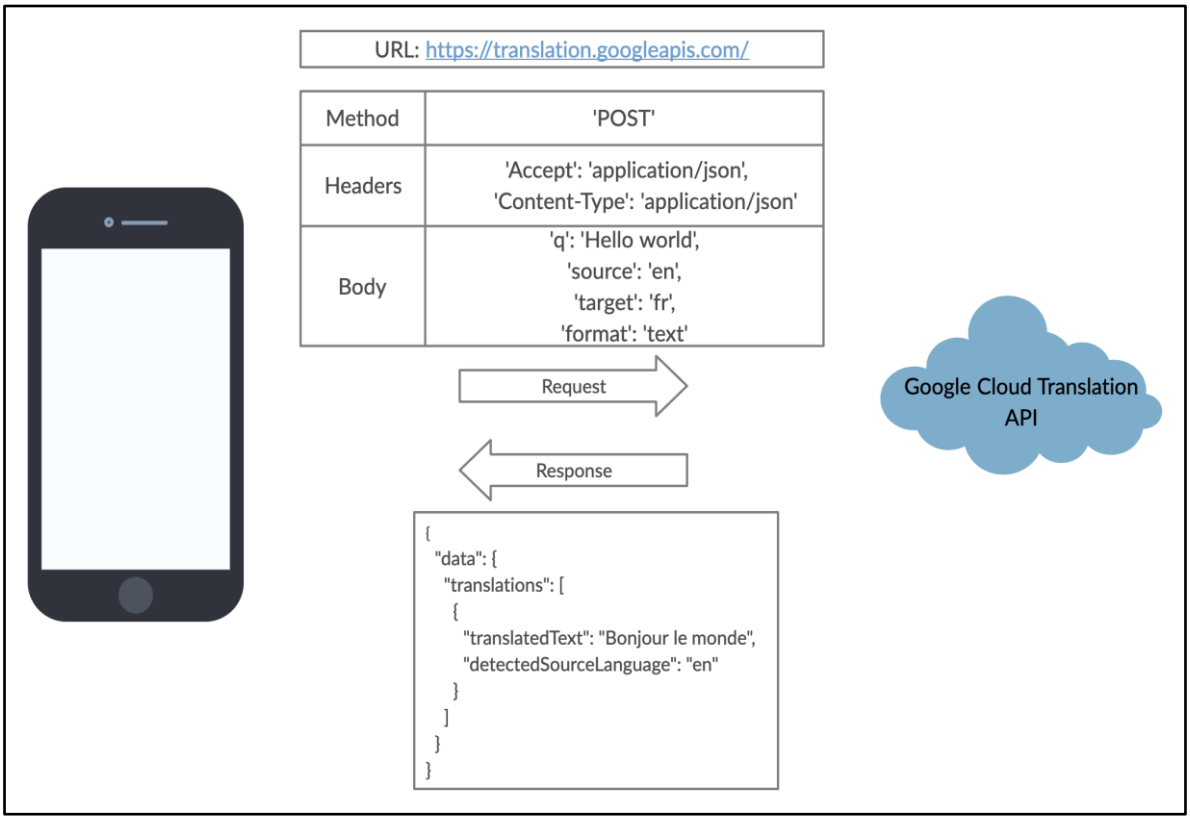


Рис. 2.9 – REST тип взаємодії з Google Translate API

Source відповідає за мову відправленого тексту, target – мову перекладу. У параметрі q необхідно вказати дані, що потребують перекладу.

## 2.5 Навігація застосунку

Навігація в проекті реалізована за допомогою бібліотеки react-navigation. Дана бібліотека реалізує три типи навігацій, які можуть бути вкладеними одна в одну або ж заміщеними:

– Stack Navigator (Стек навігація);

– Tab Navigator (Таб навігація);

– Drawer Navigator (Бокова навігація).

В розробленому застосунку використовується стек та бокова навігації. Структуру навігації зображено на Рис. 2.10.

Отже, в основі навігації закладено стек навігацію, яка складається з двох стек навігацій та однієї бокової навігації.

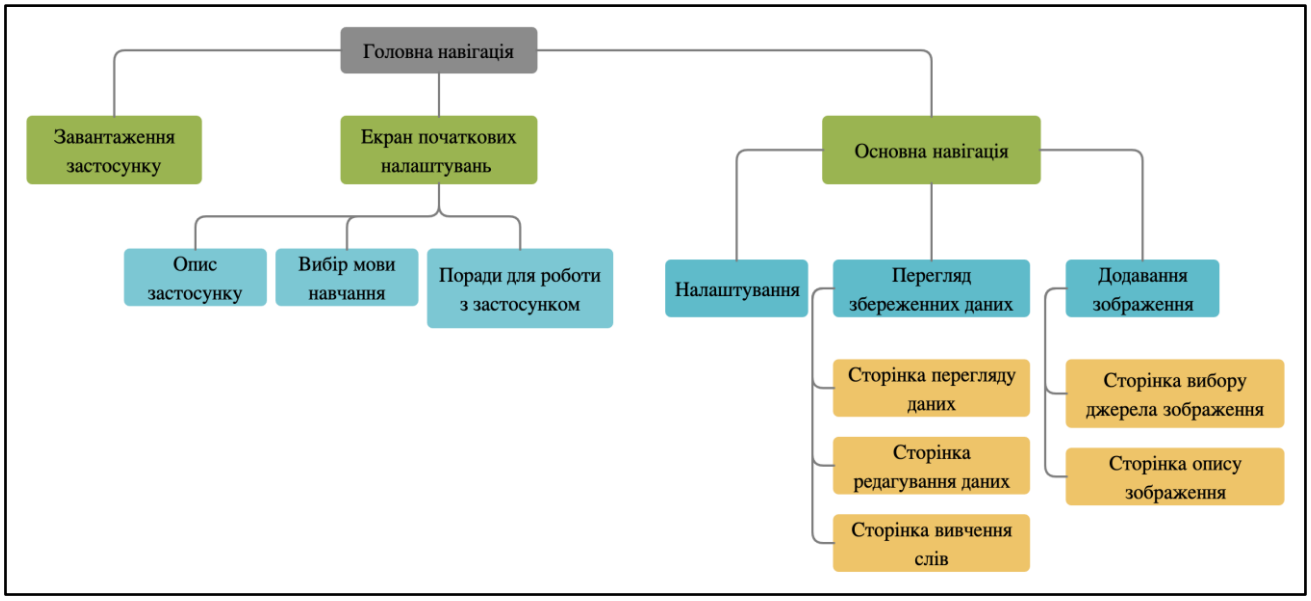


Рис. 2.10 – Структура навігації застосунку

Перша стек навігація “Завантаження застосунку” складається з одного екрану, який відображається кожен раз при завантаженні застосунку.

Друга стек навігація “Екран початкових налаштувань” викликається лише при першому завантаженні застосунку та складається з трьох екранів, які

відображають опис застосунку, вибір мови навчання та поради для роботи з застосунком.

Третя бокова навігація “Основна навігація” відображається при кожному наступному завантаженні застосунку та складається з трьох стек навігацій.

Стек навігація “Налаштування” складається з одного екрану, що містить логіку роботи з мовами для перекладу.

Стек навігація “Перегляд збережених даних” складається з трьох екранів: “Сторінка перегляду даних”, “Сторінка редагування даних” та “Сторінка вивчення слів”.

Стек навігація “Додавання зображення” містить два екрани для вибору джерела зображення та опису об’єкту зображення.

## 2.6 Модулі для роботи з даними

Важливою складовою проекту є захист користувацьких даних шляхом збереження максимальної кількості даних безпосередньо на пристрої. Це продиктовано однією з вимог проекту.

Умовно роботу з даними можна поділити на дві частини: збереження зображення в файловій системі та збереження даних в рамках застосунку.

Оскільки користувач має змогу завантажувати зображення та використовувати їх всередині застосунку, необхідно зберігати зображення в окремій папці в файловій системі, яка завжди доступна з середовища застосунку. Це потрібно також для випадків, коли користувач видаляє зображення з галереї, але бажає мати доступ до зображення в застосунку.

Для цієї цілі використано бібліотеку rn-fetch-blob. Дана бібліотека надає доступ до файлової системи з можливості редагування, створення та видалення файлів та папок. Оскільки iOS та Android платформи мають значні відмінності, то і їхня файлова структура досить різноманітна.

Нижче наведено список усіх доступних для роботи папок, а також їх належність до платформи:

- DocumentDir;
- CacheDir;
- MainBundleDir (iOS only);
- DCIMDir (Android Only);
- DownloadDir (Android Only);
- MusicDir (Android Only);
- PictureDir (Android Only);
- MovieDir (Android Only);
- RingtoneDir (Android Only);
- SDCardDir (0.9.5+ Android Only).

Оскільки існують лише дві папки, які доступні з обох мобільних платформ, було обрано папку DocumentDir для збереження всіх зображень застосунку.

Для збереження даних в рамках застосунку було обрано бібліотеку Redux. Схему її роботи наведено на Рис. 2.11.

Дана бібліотека складається з сховища даних (Store), подій (actions) та редюсерів (reducers).

Сховище даних є унікальним глобальним об’єктом, що зберігає всі дані системи. Для зміни даних в сховищі використовуються події, які мають два параметри: тип (type) та параметри (payload). Дані події потрапляють в редюсер, який обробляє подію в залежності від її типу та змінює сховище даних.

Таким чином будь-яка подія в застосунку, така як створення зображення, зміна мови чи переклад контенту супроводжується відправкою відповідної події, її обробкою в редюсері та занесенням змін у сховище.

Оскільки додаток має змогу перекладу на декілька різних мов, було обрано таку структуру сховища, згідно якої кожна мова перекладу містить окреме поле в сховищі даних redux. Для зручного доступу кожне поле мови поділене на дві складових, а саме: масив ідентифікаторів усіх існуючих зображень та безпосередньо масив самих зображень з додатковими даними.

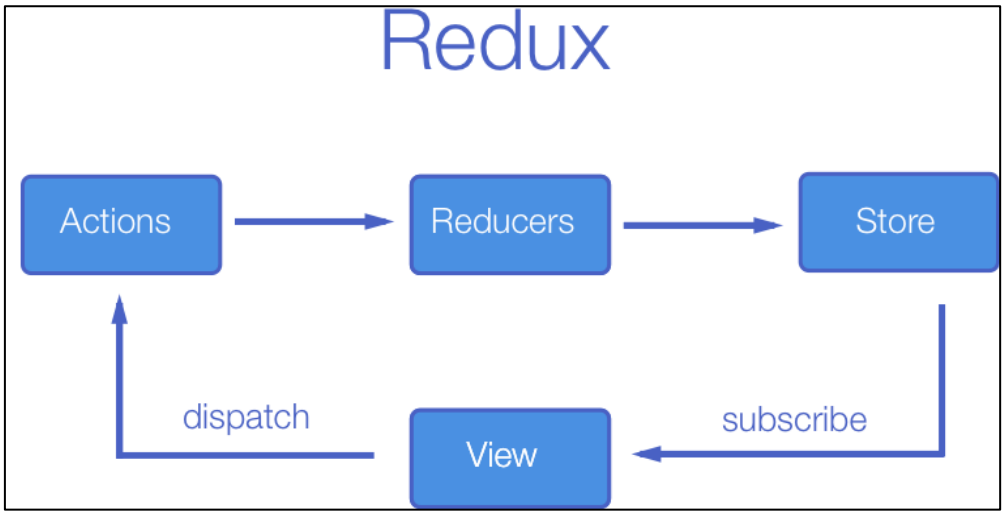


Рис. 2.11 — Структура бібліотеки Redux

Для відображення даних в компонентах використовується бібліотека reselect. Її задача полягає у підписці на сховище даних та поверненні необхідних даних. Особливістю даної бібліотеки є мемоізація даних, тобто повернення тих же даних, якщо вони не були змінені, а не нових структур, що дублікують усі поля.

Redux дозволяє зберігати дані в застосунку лише під час його роботи, тобто при закритті застосунку всі дані будуть втрачені. Для вирішення цієї проблеми доцільно використовувати бібліотеку redux-persist. Її задача полягає у збереженні даних зі сховища даних при закритті застосунку в кеш пристрою та завантажувати їх в сховище даних з кеша при відкритті застосунку.

## Висновки до розділу 2

В даному розділі розглянуто реалізацію архітектури програмного рішення.

Детально розглянуто інфраструктуру React Native, описано взаємодію Javascript коду з нативним модулем tflite-react-native для роботи з моделями нейронних мереж та Google Translate API.

Наведено реалізацію модуля tflite-react-native за допомогою C++ інтерфейсу бібліотеки TensorFlow Lite, а саме - завантаження моделей НМ та обробка даних різними моделями.

Детально описано структуру навігації застосунку, типи використаних навігацій та їх складові.

Наведено налаштування конфігурації Google Translate API для відправки запитів на переклад слів та отримання відповідей.

В останньому підрозділі описано методи роботи з даними застосунку, а саме використання бібліотек для збереження даних в файловій системі та кеші.

В результаті проведеної роботи:

1. Запропоновано програмне рішення для мобільного застосунку на основі нейронних мереж для вивчення слів іноземними мовами.
2. На основі програмного рішення розв’язано прикладну задачу визначення слів іноземними мовами за фотографією об’єкта.

# 3 МЕТОДИКА РОБОТИ З МОБІЛЬНИМ ЗАСТОСУНКОМ

Усі мобільні застосунки для iOS систем можуть бути встановлені лише з офіційного застосунку App Store. Для цього необхідно мати ідентифікатор Apple ID — обліковий запис, який використовується для різних Apple служб. Далі необхідно знайти програму у пошуку та натиснути іконку справа для завантаження [33].

Кожне відкриття застосунку супроводжується екраном завантаження (Рис. 3.1). Дана сторінка відображається в той час, поки завантажуються всі необхідні для роботи застосунку дані.


Рис. 3.1 — Екран завантаження застосунку

Наступний екран відображається в залежності від того, чи перший раз відкрито застосунок. Якщо застосунок ще не було відкрито жодного разу, то відкривається сценарій ініціалізації додатку, інакше — головний сценарій

застосунку, тобто головна навігація з обраним екраном для завантаження зображення.

Головна навігація застосунку складається з трьох частин: сценарію завантаження зображення, сценарію роботи з контентом та сценарію налаштувань.

## 3.1 Ініціалізація застосунку

Ініціалізація застосунку відбувається при першому відкритті застосунку та складається з трьох кроків — екранів. Перший екран відображається три секунди, вітаючи користувача та надаючи короткий опис застосунку (Рис. 3.2).


Рис. 3.2 — Екран привітання

Далі з’являється наступний екран, що відповідає за вибір початкових налаштувань, а саме встановлення мови для перекладу слів.

Наразі система підтримує три мови: англійську, французьку та іспанську (Рис. 3.3). Екран містить текст, що пояснює користувачу дію, яку від нього очікує

застосунок. Користувач може обрати лише одну мову, після чого з’явиться наступний екран. Також виведено повідомлення про дані, що обрана мова може бути змінена пізніше в налаштуваннях.

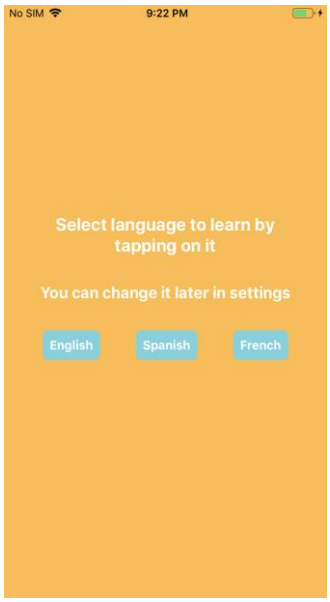


Рис. 3.3 – Екран вибору мови

Після вибору мови завантажується третій екран (Рис. 3.4).

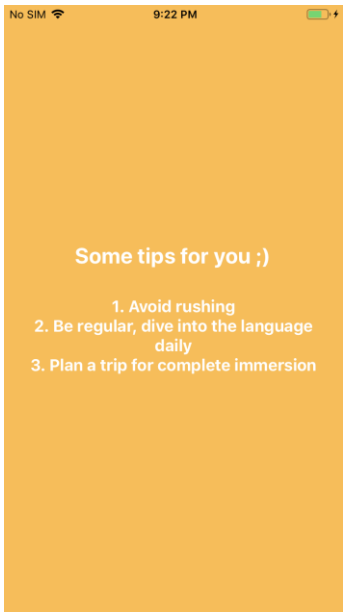


Рис. 3.4 – Екран порад для користувача

На цьому екрані виведено декілька порад для успішного вивчення слів:

– “Уникати спішки”;

– “Вивчати слова регулярно” ;

– “Спланувати подорож для повного занурення у мову вивчення”.

Даний екран відображається три секунди, після чого відкривається головна навігація застосунку, яка містить три сценарії: “завантаження зображення” (за замовчуванням), “робота з контентом” та “налаштування” застосунку.

## 3.2 Завантаження зображення

Розроблений мобільний застосунок підтримує два способи завантаження зображення: з галереї фотографій та безпосередньо з камери пристрою (Рис. 3.5) .

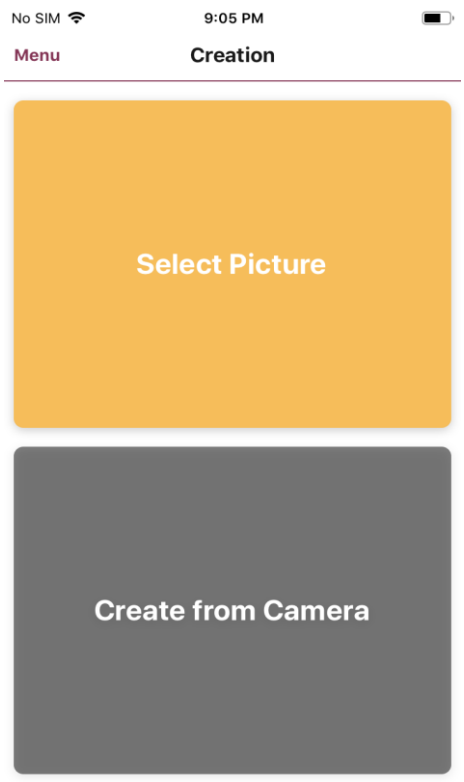


Рис. 3.5 — Екран вибору способу завантаження зображення

Отже, на екрані вибору способу завантаження користувач бачить дві великі кнопки, що відповідають за завантаження зображення з галереї та камери.

Також у верхньому лівому кутку екрану розміщується основне меню застосунку. При виборі першого способу відкривається екран вибору зображення з галереї, другого ж — екран з вбудованою в застосунок камерою.

Згідно з політикою компанії Apple перш ніж отримати доступ до певних ресурсів користувача, до яких належать галерея та камера, користувач повинен надати програмі дозвіл на доступ до особистої інформації. Це можливо завдяки появі спеціального сповіщення, який має дві опції: заборонити або ж дозволити застосунку використовувати дані користувача. Дане сповіщення повинне містити пояснення, чому застосунок потребує дозволу на використання цих даних.

### 3.2.1. Завантаження зображення з галереї

При виборі першої опції користувач має змогу обрати будь-яке зображення з галереї свого пристрою (Рис. 3.6).

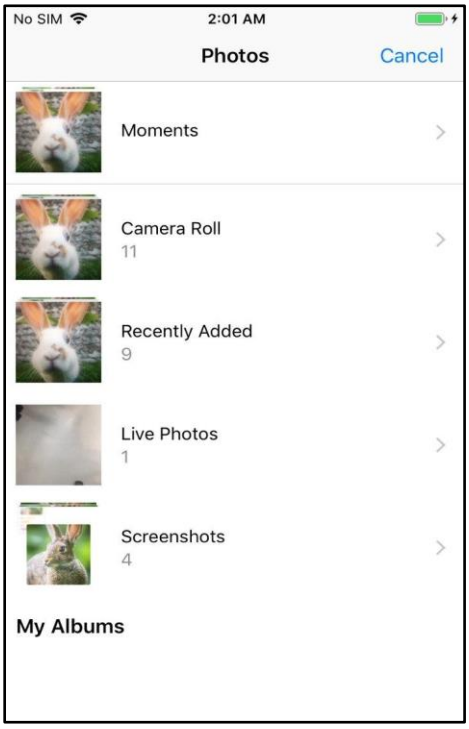


Рис. 3.6 — Екран вибору зображення із галереї

Вибране зображення автоматично завантажується в окрему папку для забезпечення постійного доступу з застосунку.

Проте перш, ніж застосунок отримає доступ до галереї, користувач повинен надати дозвіл застосунку використовувати його персональні дані. Це можливо за вибору необхідної опції в сповіщенні (Рис. 3.7) при першому відкритті галереї у застосунку.

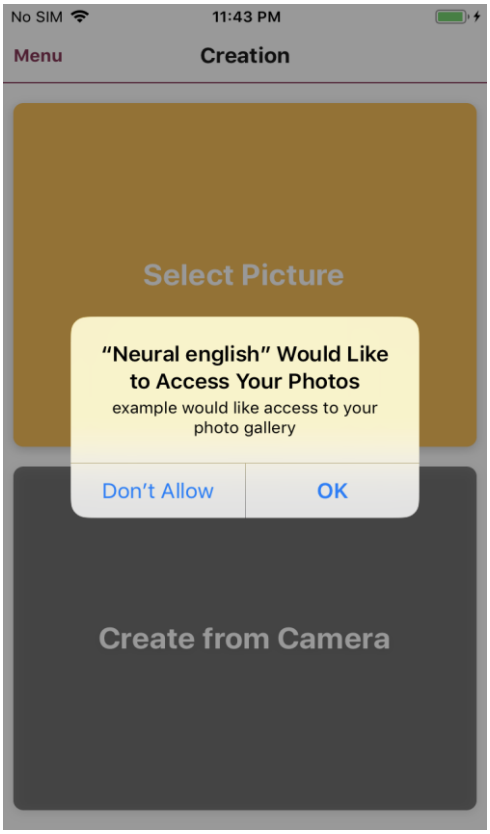


Рис. 3.7 — Сповіщення про запит доступу до галереї

Будь-яке таке сповіщення містить назву застосунку, що відправляє запит на отримання дозволу, а також пояснення, для чого цей доступ потрібен. Тільки після підтвердження доступу користувач зможе відкрити галерею безпосередньо з застосунку.

### 3.2.2. Завантаження зображення з камери

Обравши варіант завантаження зображення з камери відкривається відповідна сторінка з вбудованою камерою (Рис. 3.8). Даний екран містить в правому верхньому кутку кнопку закриття екрану.


Рис. 3.8 — Екран з вбудованою камерою

Таким чином користувач має змогу закрити екран, не створивши зображення та повернутись на екран вибору способу завантаження зображення.

По центру екрану горизонтально та в нижній частині розміщено кнопку для захоплення знімку, а безпосередньо на самому екрані відображається зображення з камери.

Таким чином зображення буде створене тільки після натискання кнопки захоплення. Дане зображення буде збережене в папці застосунку, створеній всередині папки Documents для можливості доступу з застосунку.

Та перш ніж отримати доступ до камери з застосунку, користувач повинен підтвердити дозвіл. Це можливо за допомогою спеціального сповіщення (Рис. 3.9), що з’являється при першому відкритті екрану камери. Дане сповіщення містить назву застосунку, що запитує дозвіл, тип персональних даних, які будуть використовуватись (в даному випадку – камера) та опис причини запиту.

Якщо користувач обирає опцію “ОК”, то автоматично відкривається екран камери. Обравши опцію “Don’t allow” користувач не зможе відкрити камеру в рамках застосунку.

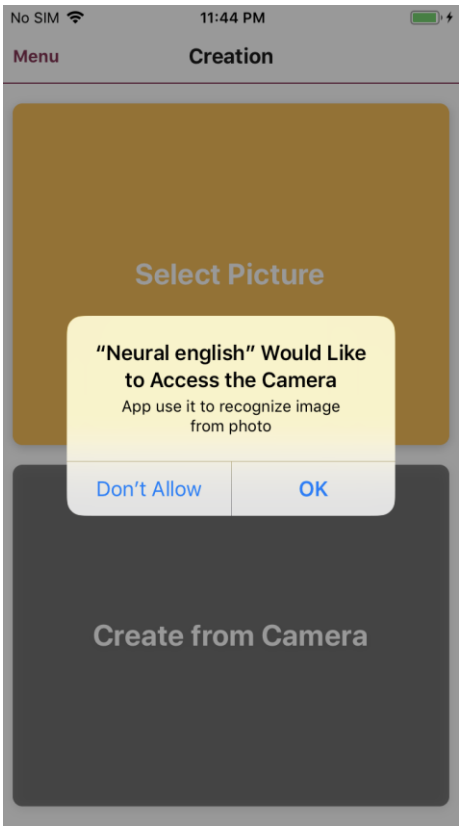


Рис. 3.9 — Сповіщення про запит доступу до камери

Проте дозволити доступ завжди можна в налаштуваннях пристрою, а саме у вкладці застосунку.

### 3.2.3. Опис розпізнаного об’єкту зображення

Обравши будь-який спосіб завантаження та отримавши зображення, користувач переходить на екран опису зображення (Рис 3.10).

Вгорі в лівому кутку розміщено кнопку Меню, що відкриває бокову навігацію застосунку.

На самому екрані розміщено створене зображення в зменшеній версії, під яким знаходиться декілька кнопок з варіантами об’єктів на зображення. Дані кнопки — це припущення об’єктів зображення, розпізнаних за допомогою нейронних мереж. При виборі будь-якого варіанту з припущень зображення з

описом та перекладом зберігається в застосунку до відображається на екрані словника.

Якщо ж жодне з припущень не задовольняє користувача або виявилось хибним, то існує можливість додавання власного опису об’єкту зображення. Це можливо завдяки спеціальному полю для ручного вводу даних, яке розміщене нижче припущень НМ.


Рис. 3.10 — Екран заповнення даних зображення

Додавши опис зображення, користувач повинен натиснути кнопку “Add to learning” для збереження зображення з описом та перекладом та його відображення на екрані словника.

## 3.3 Робота з контентом

Після успішного завантаження ряду зображень користувач має змогу переглянути всі дані у вигляді списку.

Для цього необхідно натиснути у лівому верхньому кутку на кнопку Menu та у боковій навігації обрати опцію меню Learning – це друга частина система, яка відповідає за роботу з контентом. Після чого буде відображено сторінку “Dictionary” (Рис. 3.11). У верхній частині відображаються дві кнопки: “Menu” та “Study”. Остання відкриває екран навчання, про який згодом.

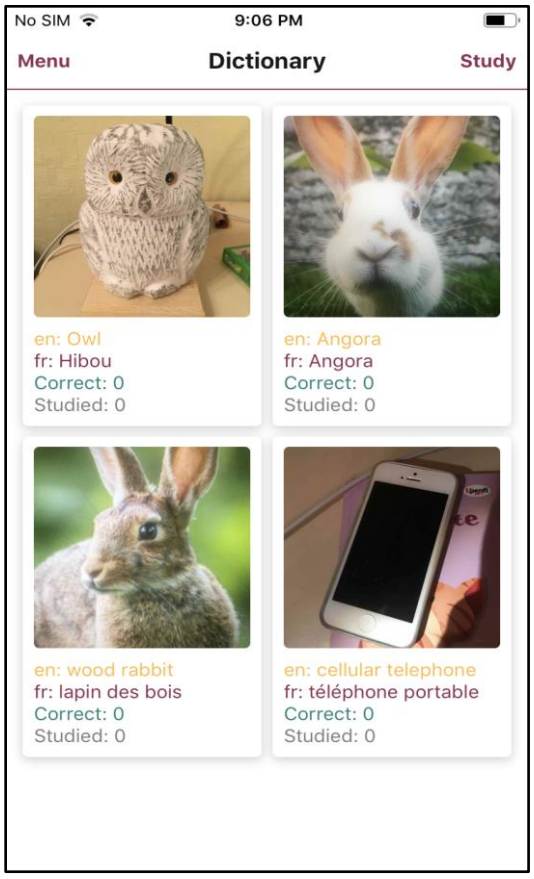


Рис. 3.11 — Екран словника

Безпосередньо на екрані відображено список всіх створених зображень з детальним описом, який складається з чотирьох рядків.

Перший рядок надає інформацію про опис зображення мовою користувача.

Другий рядок – рядок перекладу опису мовою вивчення.

Наступні два рядки відповідають за статистичну інформацію, а саме кількість спроб вивчення слова та кількість вдалих спроб.

При кліку на зображення відкривається модальне вікно з збільшеним зображенням, а також двома опціями: редагування та видалення опису зображення (Рис. 3.12).

Видалення зображення супроводжується попереджувальним сповіщенням, щоб користувач мав змогу підтвердити видалення і не втратив дані через випадковий клік (Рис. 3.13).

Після підтвердження видалення користувач залишається на екрані “Dictionary”.

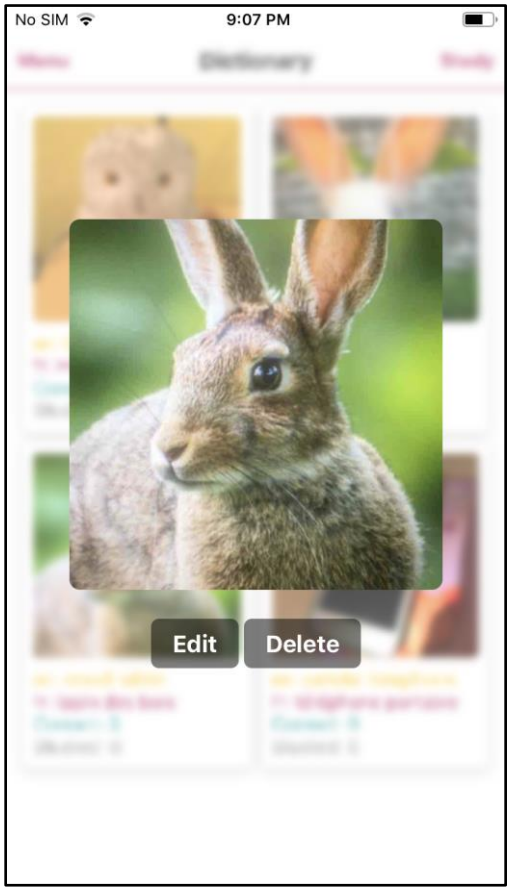


Рис. 3.12 — Екран відображення зображення у більшому розмірі

Натиснувши на опцію “Cancel” модальне вікно залишається відкритим.

Вибір опції “Edit” перенаправляє користувача на екран редагування опису.

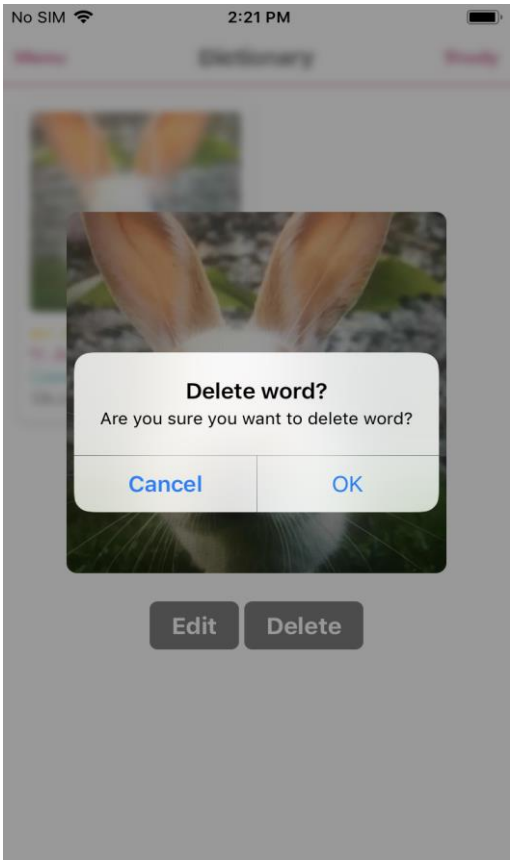


Рис. 3.13 — Сповіщення про підтвердження видалення

На цьому екрані у верхньому лівому кутку зображено кнопку “Menu”.

Основну частину сторінки займає зображення в зменшеному розмірі (Рис. 3.14). Нижче розміщено два поля вводу з підписами їх назв для зміни опису об’єкту зображення та перекладу. В даних полях за замовчуванням введені поточні дані зображення.

Для збереження внесених змін необхідно натиснути кнопку “Save”.

Редагування опису не впливає на статистичні дані зображення (такі як кількість спроб вивчення слова та кількість вдалих спроб).

Після збереження змін користувач автоматично перенаправляється на екран “Dictionary”.

Також з екрану “Dictionary” користувач може перейти на екран “Study”, де має змогу вивчити додані слова (Рис. 3.15).


Рис. 3.14 — Вікно редагування опису зображення

Якщо у користувача в словнику наявні менше 5 слів, то застосунок не зможе відкрити екран “Study” і виведе спеціальне сповіщення на екран.

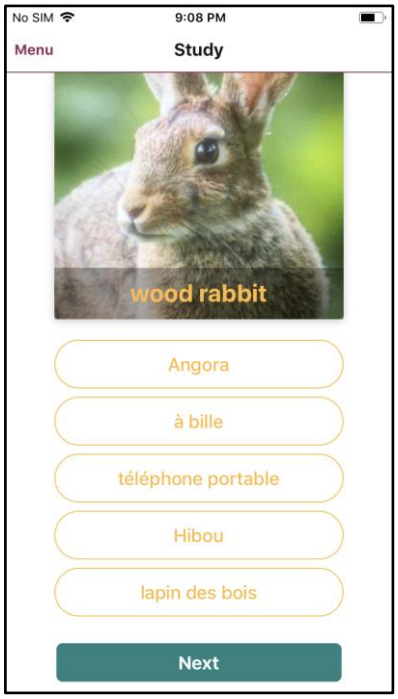


Рис. 3.15 — Екран вивчення слів

В сповіщенні описана причина відмови відкриття екрану “Study”.

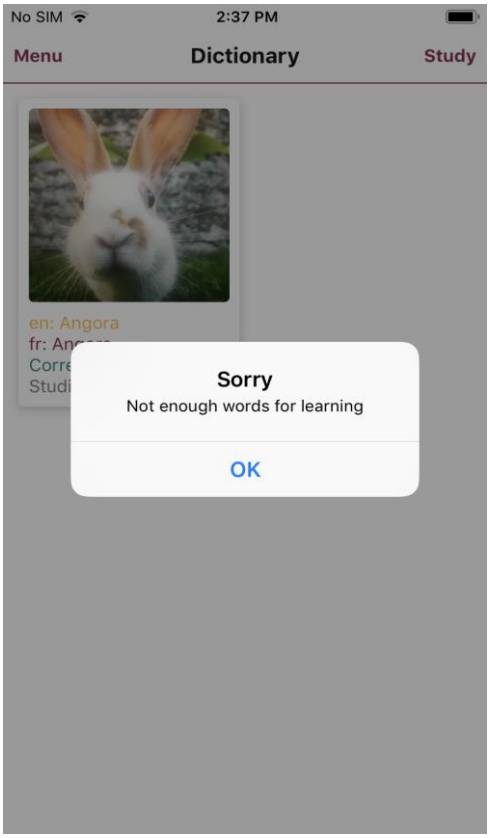


Рис. 3.16 — Сповіщення про недостатню кількість слів

Якщо в словнику достатньо даних, то відкривається екран для вивчення слів.

При відкритті даного екрану застосунку готуються для вивчення десять слів, які почергово відображаються на екрані. Зі словника беруться усі слова, що належать до обраної мови перекладу та вивчались менше десяти разів. Масив цих слів сортується в довільному порядку, після чого обираються до десяти слів.

Екран вивчення представляє собою зображення слова з його описом, а також списком із п’яти варіантів відповідей, серед яких лише одна вірна. Нижче списку відображається кнопка “Next”, яка використовується для підтвердження вибору. Таким чином унеможливлюється вибір варіанту випадковим кліком по запропонованій відповіді.

Обравши слово, система підсвічує правильний переклад зеленою рамкою, а в разі невдачі - також невірний переклад червоною рамкою (Рис. 3.17).

Користувач має змогу проаналізувати обраний варіант відповіді та вірний, тільки після чого перейти до наступного слова, натиснувши кнопку “Next”. Після завершення навчання користувач перенаправляється на екран “Dictionary”.

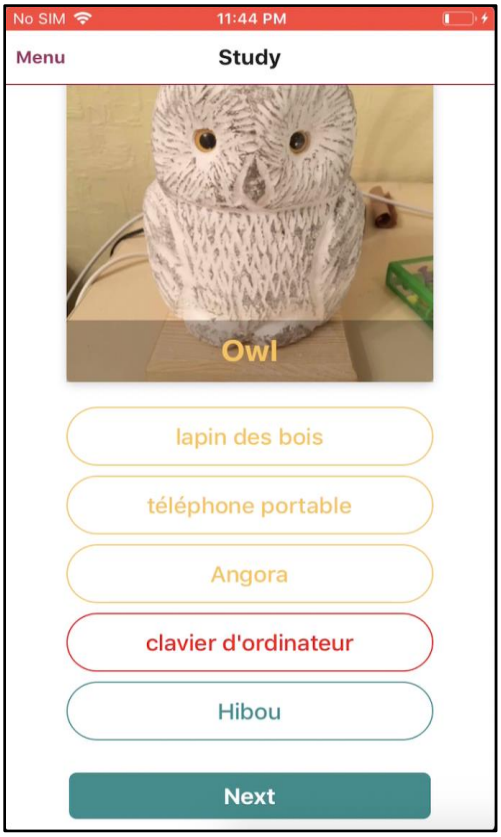


Рис. 3.17 — Екран вивчення слів після обраного варіанту перекладу

На даному екрані відображається уже оновлена статистика для кожного слова, що було обране для навчання.

## 3.4 Налаштування

Обравши в боковому меню опцію “Settings” користувач переходить на екран налаштувань (Рис. 3.18).

Даний екран містить у верхньому лівому кутку кнопку “Menu” для відкриття бокового меню.

На самому екрані відображено список з двох рядків. У першому рядку відображається інформація про обрану для перекладу мову. Другий рядок при

кліку по ньому відображає список усіх мов для перекладу, що використовуються в застосунку.

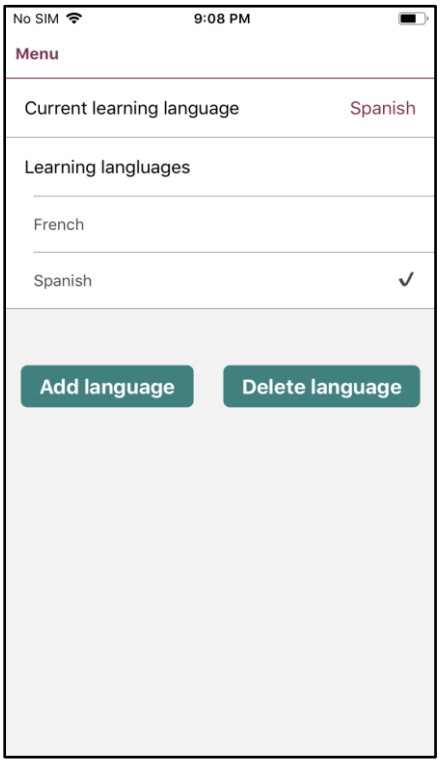


Рис. 3.18 — Екран налаштувань

Обрана мова для перекладу відображається у цьому списку з іконкою “check” у правій частині рядка.

Одночасно в користувача застосунку може бути декілька мов для навчання, кожна з яких має власний масив зображень з описом. Користувач має змогу переключати мову навчання, після чого на екрані “Dictionary” будуть відображатись додані обраною мовою слова, а також на екрані “Creation” переклад слів буде відбуватись на оновлену мову.

Якщо користувач у списку мов має менше трьох мов, в такому разі для нього доступна кнопка “Add language”. Якщо у користувача у списку мов додано більше однієї мови, то також доступна кнопка “Delete language”. Кожна кнопка відкриває модальне вікно, на якому відображено пікер з можливими варіантами вибору.

Для додавання мови модальне вікно зображене на Рис. 3.19.

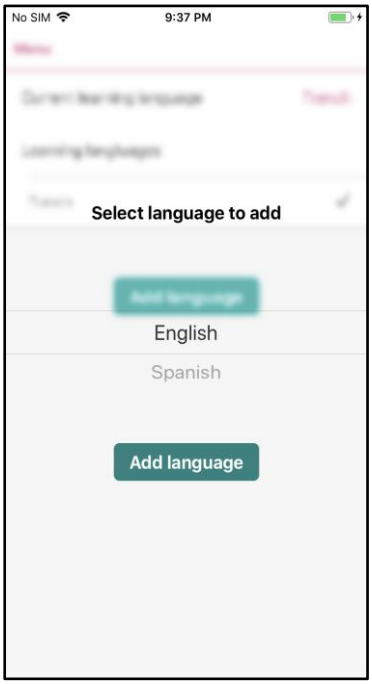


Рис. 3.19 — Модальне вікно додавання мови

Модальне вікно для видалення мови зображене на Рис. 3.20.

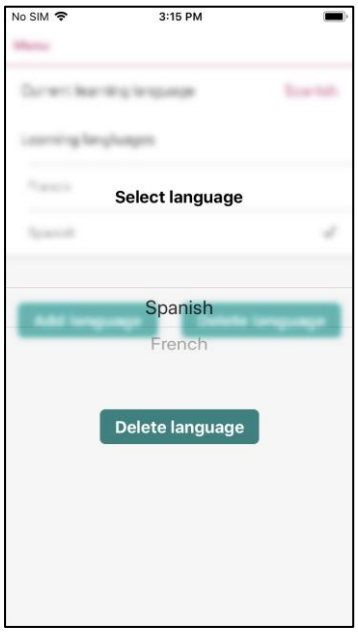


Рис. 3.20 — Модальне вікно видалення мови

Якщо користувач хоче покинути модальне вікно, то необхідно натиснути в будь-якому місці на розмитій частині екрану.

Для підтвердження необхідно обрати потрібну мову в пікері та натиснути кнопку “Save”. Після додавання обрана мова автоматично стає вибраною мовою для вивчення.

Якщо користувач вирішує видалити мову, яка наразі є обраною для навчання, то за замовчуванням мовою для навчання стає перша мова у списку доданих мов.

## Висновки до розділу 3

У даному розділі наведено інструкції для інсталяції застосунку в разі його успішного завантаження в App Store.

Описано сценарій першої ініціалізації застосунку, етапи завантаження зображення та додавання його опису.

В окремому підрозділі описано сценарій роботи з контентом, що включає в себе редагування та видалення даних, перегляд доданої інформації та статистики до неї, а також вивчення слів.

В останньому підрозділі наведено інструкції для роботи з налаштуваннями.

# 4 РОЗРОБЛЕННЯ СТАРТАП-ПРОЕКТУ

Стартап представляє собою тип бізнесу, який направлений на створення продукту чи технології та отримання доходів шляхом реалізації принципово нової ідеї. В даному розділі описано результати проведеного аналізу ринкових можливостей розроблюваного мобільного застосунку.

## 4.1 Опис ідеї проекту

Опис ідеї, основні можливі напрямки застосування з описом вигод для користувача подано у Таблиці 4.1.

Таблиця 4.1. Опис ідеї стартап-проекту

| Зміст ідеї | Напрямки застосування | Вигоди для користувача |
|---|---|---|
| Створення мобільного застосунку для вивчення іноземних слів методом зорового запам’ятовування. Ресурсами для вивчення є безпосередньо дані користувачів (зображення). | 1. Освіта | Мобільність.<br>Унікальний контент.<br>Необмеженість ресурсів.<br>Адаптація під користувача (опис зображень, мова навчання).<br>Перегляд прогресу навчання. |
| | 2. Туризм | Доступність в будь-який момент.<br>Можливість використання без інтернету.<br>Адаптація під користувача (мова навчання). |
| | 3. Індустрія дозвілля | Мобільність.<br>Корисне дозвілля.<br>Саморозвиток. |

Оскільки в кожній сфері існують системи, що мають певні схожі властивості системи, нижче подається ряд переваг даної системи (Таблиця 4.2).

Таблиця 4.2. Визначення сильних, слабких та нейтральних характеристик ідеї проекту.

| № п/п | Техніко-економічні характеристики ідеї | (потенційні) товари/концепції конкурентів | | W слабка сторона | N нейтраль-на сторона | S сильна сторона |
|---|---|---|---|---|---|---|
| | | Мій проект | Lin-gualeo | | | |
| 1 | Необхідність серверу обробки даних | +/- | + | наявна | частково наявна | відсутня |
| 2 | Необхідність доступу до інтернету | +/- | + | наявна | частково наявна | відсутня |
| 3 | Залежність від API для перекладу | + | - | наявна | відсутня | відсутня |
| 4 | Наявність веб-версії | - | + | відсутня | доступна | доступна |
| 5 | Наявність мобільної версії | + | + | відсутня | доступна | доступна |
| 6 | Можливість вибору мови навчання | + | + | відсутня | доступна | доступна |

Отже, сильними сторонами системи є:

- вибір мови навчання;
- мобільна версія;
- безпека даних.

Відповідно, слабкими сторонами є:

- відсутність веб-версії;
- Залежність від API для перекладу;
- Тренування нейронної моделі на клієнті.

## 4.2 Технологічний аудит ідеї проекту

У даному розділі проведено аудит технології для реалізації стартап-проекту (Таблиця 4.3).

Таблиця 4.3. Технологічна здійсненність ідеї проекту

| № п/п | Ідея проекту | Технології її реалізації | Наявність технології | Доступність технологій |
|---|---|---|---|---|
| 1 | Використання нейронних мереж | завантаження готової моделі на мобільний пристрій TensorFlow Lite | наявна | так |
| 2 | Використання нейронних мереж | Запит-відповідь на сервер, де зберігається мережа (TensorFlow JS) | наявна | так |
| 3 | Навчання мереж на клієнті | реалізація нейронної мережі на клієнті (TensorFlow Swift beta) | наявна, проте потребує доробки | так |
| 4 | Збереження даних на клієнті | використання доступу до файлової системи та кешу | наявна | так |
| 5 | Інструменти для перекладу тексту | використання зовнішніх інструментів (Google translate API) | наявна | так |
| 6 | Власні інструменти для перекладу тексту | Реалізація власного інструменту | відсутня | ні |

Таблиця 4.3 (Продовження)

| |
|---|
| Обрана технологія реалізації ідеї проекту:<br>– використання нейронних мереж шляхом завантаження готової моделі на мобільний пристрій (TensorFlow Lite);<br>– збереження даних на клієнті;<br>– Google translate API. |

## 4.3 Аналіз ринкових можливостей запуску стартап-проекту

Нижче проведено аналіз попиту: наявність попиту, обсяг, динаміка розвитку ринку (Таблиця 4.4).

Таблиця 4.4. Попередня характеристика потенційного ринку стартап-проекту

| No п/п | Показники стану ринку (найменування) | Характеристика |
|---|---|---|
| 1 | Кількість головних гравців, од | 2 |
| 2 | Загальний обсяг продаж, грн/ум.од | 50 тис |
| 3 | Динаміка ринку (якісна оцінка) | зростає |
| 4 | Наявність обмежень для входу (вказати характер обмежень) | немає |
| 5 | Специфічні вимоги до стандартизації та сертифікації | відповідність законам країн, в яких використовується |
| 6 | Середня норма рентабельності в галузі (або по ринку), % | 13% |

Згідно результатів таблиці ринок є привабливим для входження за попереднім оцінюванням. Нижче в Таблиці 4.5 надано вимоги споживачів до товару.

Таблиця 4.5. Характеристика потенційних клієнтів стартап-проекту

| No п/п | Потреба, що формує ринок | Цільова аудиторія (цільові сегменти ринку) | Відмінності у поведінці різних потенційних цільових груп клієнтів | Вимоги споживачів до товару |
|---|---|---|---|---|
| 1 | Автовизначення контенту зображення | Школярі, студенти, люди без знання англійської | Коректне визначення є в першу чергу важливим для школярів і студентів | коректне визначення контенту |
| 2 | Збереження даних на пристрої | Користувачі з потужними мобільними пристроями | важливе для всіх груп | ефективне використання ресурсів |
| 3 | Захист інформації (зображень) користувача | Школярі, студенти | важливе для всіх груп | захист завантаженої інформації |
| 4 | Переклад інформації | Школярі, студенти, люди без знання англійської | Обмеження на переклад небажаних слів для студентів і школярів | точний переклад |
| 5 | Вивчення слів | Школярі, студенти, люди без знання англійської | Важливо для школярів та студентів, які можуть на постійній основі використовувати систему | надійні та ефективні алгоритми для вивчення |

На основі даних про потенційні групи нижче наведено таблиці фактори загроз (Таблиця 4.6) і фактори можливостей (Таблиця 4.7).

Таблиця 4.6. Фактори загроз

| No п/п | Фактор | Зміст загрози | Можлива реакція компанії |
|---|---|---|---|
| 1 | Темпи зростання ринку | Поява конкурентів зі схожим функціоналом | Введення нового функціоналу |
| 2 | Сезонність | Туристи можуть використовувати продукт з періодичністю, з якою вони подорожують | Введення періодичних сповіщень, що запрошують користувача скористатись застосунком |
| 3 | Здатність постачальників диктувати ціни | Збільшення ціни на використання Google Translate API | Використання альтернативних інструментів |

Отже, існуючі фактори загроз можна вирішити за допомогою введення нового додаткового інструментарію та функціоналу, розвитку проекту в цілому. Основною загрозою для застосунку ж поява конкурентів.

Таблиця 4.7. Фактори можливостей

| No п/п | Фактор | Зміст можливості | Можлива реакція компанії |
|---|---|---|---|
| 1 | Розвиток TensorFlow Swift | Можливість тренувати моделі нейронної мережі на пристрої для користувачів iOS | Використання нового функціоналу для тої групи людей, для якої це можливо |
| 2 | Інструменти для перекладу | Розвиток інструментів | Періодичне дослідження нових технологій та способів їх впровадження |

Таким чином основним фактором для розвитку можливостей стартап-проекту є розвиток моделей нейронних мереж в застосунку.

Таблиця 4.8. Ступеневий аналіз конкуренції на ринку

| Особливості конкурентного середовища | В чому проявляється дана характеристика | Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспромож-ною) |
|---|---|---|
| 1. олігополістична конкуренція(тип) | На ринку існує декілька великих компаній, що надають схожі послуги | Можливе переманювання користувачів. Для уникнення цього необхідно працювати над унікальністю продукту |
| 2. За рівнем конкурентної боротьби: міжнародний | Існує декілька компаній, що забезпечують постачання схожих послуг на міжнародному ринку | Ефективний маркетинг, використання реклами |
| 3. За галузевою ознакою: міжгалузева | Міжгалузева конкуренція характерна для туризму, де користувачі можуть надати перевагу перекладачам. | Відмова від використання застосунку. Для їх збереження - вдосконалення функціоналу |
| 4. Конкуренція за видами товарів:<br>- товарно-родова | Існує загроза заміщення застосунку звичайними словниками. | Зменшення кількості використання застосунку. |
| 5. За характером конкурентних переваг: нецінова | Можливість конкурентів до швидкого розвитку власного продукту | Періодичний аналіз нових технологій та їх ефективне впровадження |
| 6. За інтенсивністю: не марочна | Важливим є якість продукту, а не бренд | Розвиток продукту |

Результати проведеного ступеневого аналізу конкуренції ринку показали, що існує декілька великих компаній-конкурентів.

Основними діями для забезпечення конкурентоспроможності є постійний і стабільний розвиток програмного продукту.

На основі результатів Таблиці 4.8 проведено аналіз конкуренції в галузі за М. Портером.

Таблиця 4.9. Аналіз конкуренції в галузі за М. Портером

| Складові аналізу | Прямі конкуренти в галузі | Потенційні конкуренти | Постачальники | Клієнти | Товари-замінники |
|---|---|---|---|---|---|
| | Lingualeo, Duolingo. | Бар'єри входження на ринок досить слабкі, оскільки на дані типи застосунків не існує патентів, товарних знаків. Також розмір капіталовкладень може бути низьким. | Фактор сили постачальників є досить сильним, оскільки існує пряма залежність від API перекладу. | Фактори сили споживачі в досить сильний, оскільки можливо досить легко замінити продукт. | Існує загроза часткового заміщення звичайними словниками, а також схожими застосунками для навчання. |
| Висновки: | Необхідно постійно працювати над розвитком додатку, оскільки існують конкуренти у галузі. | Входи в ринок досить слабкі, а це означає, що строки виходу конкурентів на ринок складають декілька місяців. | Існує часткова залежність від постачальників, безпосередньо від постачальника API перекладу. | Частково споживачі мають вплив на продукт, оскільки саме їх відгуки формують подальший розвиток продукту | Існують певні товари-замінники, проте вони не здатні повністю замінити продукт. |

Отже, як можна побачити з Таблиці 4.9, існує певна загроза з боку конкурентів, які активно працюють над власними продуктами. Тому для даного проекту є життєво-важливим слідкувати за останніми технологіями, а також розробками конкурентів та працювати над удосконаленням функціоналу і впровадженням нового.

На основі даних, отриманих в результаті аналізу характеристик ідей проекту, вимог споживачів та факторів маркетингового середовища сформовано обґрунтування факторів конкурентоспроможності (Таблиця 4.10).

Таблиця 4.10. Обґрунтування факторів конкурентоспроможності

| No п/п | Фактор конкурентоспроможності | Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим) |
|---|---|---|
| 1 | Збереження даних на пристрої | Оскільки застосунок зберігає дані на пристрої, то користувач може бути спокійним за безпеку даних, а також мати постійний доступ до них. Даного функціоналу немає в застосунках конкурентів. |
| 2 | Можливість використання без інтернету | Оскільки конкуренти використовують клієнт-серверну архітектуру, що означає необхідність в постійному стабільному доступі до інтернету, даний застосунок має перевагу завдяки можливості роботи без інтернету. |
| 3 | Розвиток технологій | Розвиток бібліотеки TensorFlow дає ряд нових можливостей для використання нейронних мереж на клієнті, що є суттєвою перевагою для застосунку. |
| 4 | Автовизначення контенту зображення | В системах клієнтів не існує можливості на основі власних зображень отримувати опис зображення, що є перевагою даного застосунку. |
| 5 | Темпи зростання ринку | Темпи зростання ринку спонукають активно та стабільно розвивати проект, оскільки доступ до нових ресурсів є спільним. |

Таблиця 4.10 (Продовження)

| | | |
|---|---|---|
| 6 | Здатність постачальників диктувати ціни | Існує певна залежність від постачальників, в даному випадку від інструментів перекладу контенту |

На основі факторів конкурентноспроможності проведено порівняльний аналіз сильних та слабких сторін проекту (Таблиця 4.11).

Таблиця 4.11. Порівняльний аналіз сильних та слабких сторін Lingualeo

| No п/п | Фактор конкуренто-спроможності | Бали 1-20 | Рейтинг товарів-конкурентів у порівнянні з Lingualeo | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | -3 | -2 | -1 | 0 | +1 | +2 | +3 |
| 1 | Збереження даних на пристрої | 15 | | + | | | | | |
| 2 | Можливість використання без інтернету | 14 | | + | | | | | |
| 3 | Розвиток технологій | 10 | | | | + | | | |
| 4 | Автовизначення контенту зображення | 10 | | | + | | | | |
| 5 | Темпи зростання ринку | 10 | | | | | | + | |
| 6 | Здатність постачальників диктувати ціни | 12 | | | | | | + | |

Порівняльний аналіз сильних та слабких сторін Lingualeo показав, що розроблений проект є слабший за такими характеристиками, як темпи зростання

ринку та залежність від постачальників. Проте має ряд сильних факторів завдяки збереженню даних на пристрої, можливості роботи без інтернету та автовизначенню контенту. Розвиток технологій не надає додаткових переваг жодному з проектів відносно одне одного.

На основі Таблиці 4.11 складено SWOT-аналіз стартап-проекту (Таблиця 4.12).

Таблиця 4.12. SWOT-аналіз стартап-проекту

| | |
|---|---|
| Сильні сторони:<br>Використання нейронних мереж.<br>Легкий та зрозумілий для вивчення інтерфейс.<br>Безпека даних користувача.<br>Відсутність прямих конкурентів. | Слабкі сторони:<br>Необхідне інвестування.<br>Неможливість навчання моделі нейронної мережі під конкретного користувача.<br>Існування застосунків, що частково можуть замінити функціонал. |
| Можливості:<br>Відносно вільний ринок.<br>Відсутність прямих аналогів.<br>Покращення алгоритму для навчання.<br>Покращення використання нейронних мереж. | Загрози:<br>Укріплення конкурентів.<br>Низький поріг входження у ринок.<br>При укріпленні конкурентів - зменшення попиту. |

SWOT-аналіз умовно поділив проект на чотири категорії: а саме: сильні, слабкі сторони проекту та можливості і загрози для виконання проекту. Отже до сильних сторін проекту можна віднести використання НМ, безпеку даних користувача та відсутність конкурентів. Проте існують слабкі сторони, які потребують доробки. До таких можна віднести неможливість навчання моделі НМ та наявність конкурентів зі схожим функціоналом.

Для покращення застосунку можливо покращувати алгоритми навчання та використання НМ.

До основної загрози можна віднести появу конкурентів через низький поріг входження.

За Таблицею 4.12 визначено альтернативи ринкового впровадження стартап-проекту, наведені у Таблиці 4.13.

Таблиця 4.13. Альтернативи ринкового впровадження стартап-проекту

| No п/п | Альтернатива (орієнтовний комплекс заходів) ринкової поведінки | Ймовірність отримання ресурсів | Строки реалізації |
|---|---|---|---|
| 1 | Розробка застосунку для вивчення іноземних слів з використанням нейронних мереж. | 80% | 1 рік |
| 2 | Розробка веб-версії визначеного застосунку | 60% | 1,5 - 2 роки |
| 3 | Розробка словника | 20% | 3 роки |

В результаті SWOT-аналізу стартап-проекту було розроблено альтернативи ринкової поведінки та час їх реалізації (Таблиця 4.13). Як висновок було обрано варіант розробки “Розробка застосунку для вивчення іноземних слів з використанням нейронних мереж”, оскільки даний варіант забезпечує найбільш ймовірне отримання ресурсів, а також є найшвидшим за строками реалізації.

## 4.4 Розроблення ринкової стратегії проекту

Розроблення ринкової стратегії передбачає вибір цільових груп потенційних споживачів (Таблиця 4.14).

Таблиця 4.14. Вибір цільових груп потенційних споживачів

| No п/п | Опис профілю цільової групи потенційних клієнтів | Готовність споживачів сприйняти продукт | Орієнтовний попит в межах цільової групи (сегменту) | Інтенсивність конкуренції в сегменті | Простота входу в сегмент |
|---|---|---|---|---|---|
| 1 | Учні | готові | високий | невелика | легко |

Таблиця 4.14. (Продовження)

| | | | | | |
|---|---|---|---|---|---|
| 2 | Туристи, Туризм | готові | високий | невелика | відносно легко |
| 3 | Сфера дозвілля | готові | низький | велика | важко |
| Які цільові групи обрано: учні(школярі/студенти), туристи | | | | | |

В результаті обраних цільових груп, таких як школярі/студенти та туристи визначено базову стратегію розвитку (Таблиця 4.15).

Таблиця 4.15. Визначення базової стратегії розвитку

| No п/п | Обрана альтернатива розвитку проекту | Стратегія охоплення ринку | Ключові конкуренто-спроможні позиції відповідно до обраної альтернативи | Базова стратегія розвитку |
|---|---|---|---|---|
| 1 | Розробка застосунку для вивчення іноземних слів з використанням нейронних мереж. | Вибірковий розподіл | Безпека даних, зростання функціоналу. | Стратегія диференці-ації |

В результаті визначення базової стратегії вибрано стратегію диференціації, яка передбачає надання продукту особливих властивостей залежно від потреб користувача, які роблять товар відмінним від товарів конкурентів. Інструментом реалізації стратегії диференціації є ринкове позиціонування. Стратегія охоплення ринку – вибірковий розподіл, який застосовується до товарів, які користуються попитом у споживачів.

Далі було обрано стратегію конкурентної поведінки (Таблиця 4.16).

Таблиця 4.16. Визначення базової стратегії конкурентної поведінки

| No п/п | Чи є проект «першопрохідцем» на ринку? | Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів? | Чи буде компанія копіювати основні характеристики товару конкурента, і які? | Стратегія конкурентної поведінки |
|---|---|---|---|---|
| 1 | В певній мірі. Існують застосунки, які мають схожий функціонал, проте повного аналогу поки не існує. | Пошук нових користувачів з частковим забиранням існуючих у конкурентів. | Головна увага буде сконцентрована на покращенні властивостей проекту, в той ще час можлива розробка веб-версії. | Стратегія заняття конкурентної ніші. |

Таблиця 4.17. Визначення стратегії позиціонування

| No п/п | Вимоги до товару цільової аудиторії | Базова стратегія розвитку | Ключові конкуренто-спроможні позиції власного стартап-проекту | Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових) |
|---|---|---|---|---|
| 1 | Безпека даних, робота без доступу до інтернету, навчання | Стратегія диференціації | Покращення алгоритму навчання як користувачів, так і моделей нейронних мереж | Навчання, безпека даних, нейронні мережі |

Згідно Таблиці 4.16 було обрано стратегію заняття конкурентної ніші як стратегії конкурентної поведінки. Дана поведінка обрана оскільки вибрана ніша є в основному непривабливою для конкурентів, а також може бути стабільною.

Основними вимогами до застосунку цільовою аудиторії є безпека даних, робота без доступу до інтернету та можливість навчання, тому було обрано стратегію диференціації, а ключовою позицією є покращення алгоритмів навчання, який дозволить використати оптимізацію. Ключові асоціації – навчання, безпека даних, нейронні мережі.

## 4.5 Розроблення маркетингової програми стартап-проекту

Для формування маркетингової концепції товару необхідно підсумувати результати конкурентоспроможності.

Таблиця 4.18. Визначення ключових переваг концепції потенційного товару

| No п/п | Потреба | Вигода, яку пропонує товар | Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити |
|---|---|---|---|
| 1 | Безпека даних | Збереження даних на пристрої користувача | Особливість, якої немає у конкурентів |
| 2 | Робота без доступу до інтернету | Можливість роботи без інтернету для всього функціоналу, крім додавання контенту | Відсутня у конкурентів |
| 3 | Навчання | Алгоритми для вивчення слів | Потребує доробки |

Формування маркетингової концепції товару визначило безпеку даних, роботу без доступу до інтернету та навчання як ключі переваги.

Нижче розроблено трирівневу модель товару (Таблиця 4.19).

Таблиця 4.19. Опис трьох рівнів моделі товару

<table>
<tr><td>Рівні товару</td><td colspan="3">Сутність та складові</td></tr>
<tr><td>I. Товар за задумом</td><td colspan="3">Мобільний застосунок для вивчення іноземних слів методом зорового запам’ятовування з акцентом на безпеці інформації та можливості навчання без доступу до інтернету.</td></tr>
<tr><td rowspan="5">II. Товар у реальному виконанні</td><td>Властивості/характеристики</td><td>М/Нм</td><td>Вр/Тх /Тл/Е/Ор</td></tr>
<tr><td>1. Безпека даних<br>2. Використання нейронних мереж</td><td>М<br>Нм</td><td>Тх<br>Тл</td></tr>
<tr><td colspan="3">Якість: безпечний, легкий інтерфейс,</td></tr>
<tr><td colspan="3">Пакування: мобільний застосунок, доступний з AppStore</td></tr>
<tr><td colspan="3">Марка: NeuralEnglish</td></tr>
<tr><td rowspan="2">III. Товар із підкріпленням</td><td colspan="3">До продажу: не потребує особливих вимог</td></tr>
<tr><td colspan="3">Після продажу: не потребує особливих вимог</td></tr>
<tr><td colspan="4">За рахунок чого потенційний товар буде захищено від копіювання:</td></tr>
</table>

Захист товару буде здійснено за рахунок захисту інтелектуальної власності. Особливих вимог до використання немає. Далі необхідно встановити межі допустимих цін (Таблиця 4.20).

Таблиця 4.20. Визначення меж встановлення ціни

| No п/п | Рівень цін на товари-замінники | Рівень цін на товари-аналоги | Рівень доходів цільової групи споживачів | Верхня та нижня межі встановлення ціни на товар/послугу |
|---|---|---|---|---|
| 1 | 50грн/місяць | відсутні | 7-10 тис грн | 25-40грн/місяць |

Після визначення меж встановлення ціни було визначено систему збуту (Таблиця 4.21), а саме - збут через інтернет.

Таблиця 4.21. Формування системи збуту

| No п/п | Специфіка закупівельної поведінки цільових клієнтів | Функції збуту, які має виконувати постачальник товару | Глибина каналу збуту | Оптимальна система збуту |
|---|---|---|---|---|
| 1 | Закупівля відбувається через інтернет | -пошук користувачів<br>-статистика<br>-реклама | Канал нульового рівня | Через інтернет |

Останнім кроком маркетингової програми є розроблення концепції маркетингових комунікацій (Таблиця 4.22), що опирається на попередні визначені дані.

Таблиця 4.22. Концепція маркетингових комунікацій

| No п/п | Специфіка поведінки цільових клієнтів | Канали комунікацій, якими користуються цільові клієнти | Ключові позиції, обрані для позиціонування | Завдання рекламного повідомлення | Концепція рекламного звернення |
|---|---|---|---|---|---|
| 1 | Користувачі дізнаються про новий функціонал завдяки сповіщенням та рекламі. | Соціальні мережі | Інтернет-маркетинг | Презентація застосунку та його виключних особливостей | “Вивчення іноземної мови 24/7” |

В якості ключових позицій було обрано інтернет-маркетинг. Завданням рекламного повідомлення є презентація застосунку та його виключних особливостей.

## Висновки до розділу 4

В результаті проведеного аналізу ринкових можливостей розроблюваного застосунку розроблено опис ідеї проекту, визначено слабкі та сильні сторони. Проведено технічний аудит, в результаті якого обрано TensorFlow Lite та Google Translate API для реалізації проекту. Завдяки аналізу ринкових можливостей визначено конкурентоспроможність застосунку, цільову аудиторію. В результаті визначення базової стратегії вибрано стратегію диференціації. Стратегія охоплення ринку – вибірковий розподіл, який застосовується до товарів, які користуються попитом у споживачів. Цільовою аудиторією є студенти/школярі та туристи. Визначено, що для даного проекту є життєво-важливим слідкувати за останніми технологіями, а також розробками конкурентів та працювати над удосконаленням функціоналу і впровадженням нового.

# ВИСНОВКИ

В результаті виконаної роботи було здобуто знання про роботу нейронних мереж, визначено існуючі типи нейронних мереж та способи їх конфігурації.

Проведено аналіз проблеми використання нейронних мереж в мобільних застосунках, систематизовано існуючі методи реалізації. Здійснено аналіз можливих інструментів для виконання завдання. На основі обраної теми відібрано застосунки, що мають схожі функції, проведено детальний аналіз їх створення та сфери застосування. Основним інструментом для роботи з нейронними мережами обрано TensorFlow Lite, вивчено його архітектуру та методи використання.

Вивчено роботу фреймворку React Native, завдяки якому реалізовано інтерфейс програми та роботу з модулем, що реалізує виконання моделі нейронної мережі.

Безпосередньо в застосунку реалізовано повноцінну систему, що включає в себе модуль для запуску нейронних мереж, інструменти для роботи з даними застосунку, а також навігацію. Для даних цілей використовувались як сторонні загально відомі бібліотеки для вирішення конкретних типів задач, так і реалізовані власні алгоритми та логіка проекту. Для виконання перекладу контенту було успішно підключено Google Translate API.

В результаті проведеної роботи поглиблено навички роботи з нейронними мережами, мовою програмування JavaScript, а також з фреймворками TensorFlow Lite та react-native. Використання фреймворку react-native дало змогу реалізувати інтерфейс з мінімальною кількістю змін в залежності від мобільної платформи.

Додатково отримано навички роботи з нативним кодом, реалізованим мовою програмування Swift.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. Hosted models. [Електронний ресурс] – Режим доступу до ресурсу: https://www.tensorflow.org/lite/guide/hosted_models
2. Andrew G Howard: Efficient convolutional neural networks for mobile vision applications / Andrew G Howard, Menglong ZhuMobilenets / 2017
3. Jürgen Schmidhuber. 2015. Deep learning in neural networks: An overview. Neural networks 61 (2015), 85–117.
4. Javier Gonzalez-Dominguez. Frameby-frame language identification in short utterances using deep neural networks. Neural Networks, 64:49–58, 2015.
5. Benjamin Recht. A lock-free approach to parallelizing stochastic gradient descent. In Advances in Neural Information Processing Systems. / Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. Hogwild / 2011. 693–701.
6. A flexible and efficient library for deep learning [Електронний ресурс] – Режим доступу до ресурсу: https://mxnet.apache.org/
7. Apache MXNet на AWS.[Електронний ресурс] – Режим доступу до ресурсу:: https://aws.amazon.com/ru/mxnet
8. PyTorch Mobile [Електронний ресурс] – Режим доступу до ресурсу: https://pytorch.org/mobile/home/
9. PyTorch Mobile: Exploring Facebook’s new mobile machine learning solution [Електронний ресурс] – Режим доступу до ресурсу: https://heartbeat.fritz.ai/pytorch-mobile-exploring-facebooks-new-mobile-machine-learning-solution-96c99efbfd58
10. TensorFlow Lite guide. [Електронний ресурс] – Режим доступу до ресурсу:: https://www.tensorflow.org/lite/guide
11. Как нейросети и машинное обучение используются в мобильных устройствах и приложениях. [Електронний ресурс] – Режим доступу до ресурсу: https://www.likeni.ru/analytics/kak-neyroseti-i-mashinnoe-obuchenie-ispolzuyutsya-v-mobilnykh-ustroystvakh-i-prilozheniyakh/

12. Using TensorFlow Lite on Android [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/tensorflow/using-tensorflow-lite-on-android-9bbc9cb7d69d
13. The Essential Guide To Learn TensorFlow Mobile and Tensorflow Lite [Електронний ресурс] – Режим доступу до ресурсу: https://towardsdatascience.com/the-essential-guide-to-learn-tensorflow-mobile-and-tensorflow-lite-a70591687800
14. Google представила библиотеку TensorFlow Lite [Електронний ресурс] – Режим доступу до ресурсу: https://tproger.ru/news/google-released-tensorflow-lite/
15. How TensorFlow Lite Optimizes Neural Networks for Mobile Machine Learning [Електронний ресурс] – Режим доступу до ресурсу: https://heartbeat.fritz.ai/how-tensorflow-lite-optimizes-neural-networks-for-mobile-machine-learning-e6ffa7f8ee12
16. FlatBuffers [Електронний ресурс] – Режим доступу до ресурсу: https://google.github.io/flatbuffers/
17. MobileNets: Open-Source Models for Efficient On-Device Vision [Електронний ресурс] – Режим доступу до ресурсу: https://ai.googleblog.com/2017/06/mobilenets-open-source-models-for.html
18. Richard Kho. React Native by Example: Packt Publishing/ 2017, 58–414.
19. Akshat Paul. React Native for Mobile Development / Akshat Paul, Abhishek Nalwaya: Apress; 2nd ed. Edition/ 2019, 14–256.
20. Dan Ward. React Native Cookbook - Second Edition: Packt (019), 36–592.
21. Bonnie Eisenman. Learning React Native: Building Native Mobile Apps with JavaScript 2nd Edition/ 2017, 54–242.
22. Nader Dabit. React Native in Action: Developing iOS and Android Apps with JavaScript. /2019, 224–320.
23. Adam Boduch. React and React Native - Second Edition /2018, 73–540.

24. Devin Abbott. Fullstack React Native: Create beautiful mobile apps with JavaScript and React Native / Devin Abbott, Houssein Djirdeh, Anthony Accomazzo and Sophia Shoemaker /2019, 200–688.
25. Frank Zammetti. Practical React Native: Build Two Full Projects and One Full Game using React Native./Apress, 2018, 100–334.
26. Emilio Rodriguez Martinez. React: Cross-Platform Application Development with React Native: Build 4 real-world apps with React Native./Packt, 2018, 32–182.
27. What Happens When my React Native Application Starts? [Електронний ресурс] – Режим доступу до ресурсу: https://levelup.gitconnected.com/wait-what-happens-when-my-react-native-application-starts-an-in-depth-look-inside-react-native-5f306ef3250f
28. React Native Tools [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/applantic/react-native-tools-396c5a0190a3
29. 6 Tools for Debugging React Native [Електронний ресурс] – Режим доступу до ресурсу: https://www.sitepoint.com/tools-for-debugging-react-native/
30. Otavio Good. How Google Translate squeezes deep learning onto a phone, 2015.
31. Translating text (Basic) [Електронний ресурс] – Режим доступу до ресурсу: https://cloud.google.com/translate/docs/basic/translating-text?hl=UK
32. Как задеплоить нейронную сеть на мобильном. [Електронний ресурс] – Режим доступу до ресурсу: https://spalah.com/blog/kak-zadeploity-neyronnuyu-sety-na-mobilynom-i-drugie-doklady-kharkov-ai-club

# ДОДАТОК А

Реалізація нейронних мереж в мобільних застосунках

УКР.НТУУ”КПІ ім. Ігоря Сікорського” _ТЕФ_АПЕПС_ ТВ42128_19М

Аркушів 1

Київ 2019

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**
**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**
**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**
**ІМЕНІ ІГОРЯ СІКОРСЬКОГО»**

# СУЧАСНІ ПРОБЛЕМИ НАУКОВОГО ЗАБЕЗПЕЧЕННЯ ЕНЕРГЕТИКИ

Матеріали XVII Міжнародної науково-практичної конференції молодих вчених та студентів м. Київ, 23-26 квітня 2019 року,

ТОМ 2

**Київ- 2019**

**УДК 004.032.26**

Магістрант 1 курсу, гр. ТВ-81мп Марич Т.І.
Доц., к.т.н. Шаповалова С.І.

## ЗАСТОСУВАННЯ НЕЙРОННИХ МЕРЕЖ В МОБІЛЬНИХ ЗАСТОСУНКАХ

За останні декілька років нейронні мережі здобули велику популярність. Це пов'язано з широким спектром їх використання: розпізнавання і обробка об'єктів зображень, розпізнавання та відтворення мови, пошук найкращих пропозицій для клієнтів в веб-застосунках. Широке поширення нейромережі отримали і в розробці мобільних додатків. Існує великий спектр їх використання, наприклад, для розпізнавання людей і створення унікальних зображень, в клавіатурі IOS систем, яка видає припущення щодо слова, що набирається. Також деякі пристрої мають систему розпізнавання схожих людей на фотографіях і сортування таких фотографій в окремі папки. Звісно, перелік використання значно ширший. Тому створення архітектурного рішення розв`язання таких задач є актуальним і має практичне значення.

Варто враховувати, що всі ці програми нічому не вчаться безпосередньо у користувача. Це пов'язано з низьким рівнем потужності сучасних мобільних телефонів. Тому найчастіше весь процес навчання відбувається на віддаленому сервері з великою кількістю GPU. Отже, існує два способи використання нейронних мереж в мобільних застосунках: шляхом відправлення запитів на сервер або ж використанням моделі безпосередньо на пристрої. Перший спосіб характеризується постійним навчанням моделі за рахунок запитів користувачів. Наприклад, на стороні клієнта задане вхідне зображення. Це зображення відправляється на сервер, де відбувається аналіз за допомогою моделі та відправлення результату клієнту. Перевагами другого способу є конфіденційність даних, незалежність роботи від наявності інтернету, а в майбутньому - і автоматичне оновлення системи за рахунок самонавчання. Популярним рішенням є використання Tensorflow Lite.

Tensorflow Lite - легковісна бібліотека, орієнтована на розробників мобільних та вбудованих пристроїв для розробки додатків під Android, iOS, Raspberry PI та інших. Найважчою частиною використання бібліотеки є підготовка готової моделі, а саме конвертація її в модель .tflite. Перевагою даної бібліотеки є набір готових до використання моделей для різних типів задач:

- MobileNet для класифікації зображень, а саме - ідентифікації сотень типів об'єктів;
- PoseNet для розпізнавання пози людини на зображенні або відео;
- Coco-ssd для розпізнавання кількох об'єктів на зображенні з обмежувальними рамками (80 різних класів об'єктів);
- Smart reply для створення пропозицій відповідей на основі повідомлень у чаті;
- DeepLab для сегментації семантичного зображення.

В роботі було визначено базові моделі навчання, які можна використовувати для мобільних за стосунків.

Перелік посилань:

1. Training and Serving ML models with tf.keras. URL: https://medium.com/tensorflow/training-and-serving-ml-models-with-tf-keras-fd975cc0fa27
2. Hosted models. URL: https://www.tensorflow.org/lite/guide/hosted_models